Некоммерческое акционерное общество
«АЛМАТИНСКИЙ УНИВЕРСИТЕТ ЭНЕРГЕТИКИ И СВЯЗИ»

Кафедра «Автоматическая электросвязь»
Специальность 6М071900 «Радиотехника, электроника и телекоммуникации»

ДОПУЩЕН К ЗАЩИТЕ
Зав. кафедрой
Чежимбаева К.С.
«____» января 2014 г.

МАГИСТЕРСКАЯ ДИССЕРТАЦИЯ
пояснительная записка

на тему: Разработка многоканального измерителя электропроводимости

Выполнил магистрант гр. ССЗп–12–1 _____ Хамзина Д.Е.
(подпись) (Ф.И.О.)

Руководитель доктор физико-химических наук, профессор _____ Шалтыкова Д.
(ученая степень, звание) (подпись) (Ф.И.О.)

Алматы, 2014

Аннотация

В данной работе описываается программно-аппаратный комплекс для измерения пространственного распределения электропроводности раствора, обеспечивающий последующую математическую обработку результатов измерений программными средствами. Успешно проведена экспериментальная работа при изучения динамических процессов, сопутствующих эффекту Джумадилова.

Аңдатпа

Берілген жұмыста программалық жолмен өлшеу нәтижелерді тізбектей математикалық өңдейтін сұйықтықтың электроөткіштігін кеңістіктік таралуын өлшеу үшін арналған аппаратты-программалық кешен ұсынылады. Кешен көмегімен Джумадилов әсеріне сай динамикалық процесстерді зерттеу жолында сәтті тәжірибе өткізілді.

Содержание

Введение

Кондуктометр, это прибор, который измеряет электропроводность электролита. Кондуктометры применяются для анализа твердых веществ, растворов водных и неводных, коллоидных систем и различных расплавов. Контроль определенного качества вод, паров или конденсатов возможен с помощью измерения кондуктометром удельной проводимости и удельного сопротивления.

Для изучения физико-химических характеристик проводящих жидкостей в настоящее время широкое распространение получили кондуктометры различных модификаций. Однако для изучения динамических процессов в таких средах часто не достаточно регистрировать проводимость в отдельных точках раствора. Примером являются исследования явлений переноса протона от геля донора к гелю-акцептору при дистанционном взаимодействии сшитых сеток. Такого рода процессы являются быстропротекающими, что требует высокой степени синхронизации при измерении электропроводности в некоторой совокупности точек раствора. В данной работе предложен программно-аппаратный комплекс для измерения пространственного распределения электропроводности раствора, обеспечивающий последующую математическую обработку результатов измерений программными средствами.

Разработаны модификации принципиальных радиоэлектронных схемы программно-аппаратного комплекса, обеспечивающего проведение измерений пространственно-временного распределения электропроводности раствора, осуществлено их макетирование и выполнено сопутствующее программное обеспечение в альфа-версии. Проведено тестирование готового изделия, доказана его пригодность для изучения динамических процессов, сопутствующих эффекту Джумадилова.

1 Разработка оборудования для изучения процессов сорбции низкомолекулярных солей интергелевыми системами.

1.1 Анализ существующих кондуктометров

Структурная схема кондуктометра**,** как правило, бывает однотрансформаторной, двухтрансформаторной или многообмоточной. В результате проведенных научных исследований было установлено, что однотрансформаторные схемы, использующие прямой метод измерения, желательно использовать из-за своей простоты и надежности как датчики-сигнализаторы. Диапазон производимых ими измерений относительно небольшой (0.1-10 См/м). Двухтрансформаторные кондуктометры работают в более широком диапазоне (0.1-100 См/м. Наибольшее распространение из-за ряда очевидных преимуществ при создании микропроцессорных кондуктометров получили многообмоточные трансформаторные схемы. Они практически не имеют в точке равновесия зависимости чувствительности от частоты напряжения питания и не требуют подключать дополнительные устройства для того, чтобы определить разбаланс.

Изучив различные варианты, по которым может строиться схема кондуктометра в трансформаторном исполнении, можно сделать нижеследующие выводы.

Погрешность измерений зависит от выбора схемы кондуктометра

Однотрансформаторная схема кондуктометра, использующая прямой метод измерения (измерительная обмотка одновременно служит и питающей, поэтому чувствительность по всему диапазону измерений не может постоянной), дает довольно высокую погрешность. Связано это с создаваемыми наводками, нестабильностью напряжения питания и т.д. Поэтому применять такие схемы для точных измерений удельной электропроводности с погрешностью менее 10 процентов не рекомендуется. Вдобавок стоит отметить, погрешность эта отнюдь не является величиной постоянной для всего диапазона измерений.

Двухтрансформаторные кондуктометры связываются между собой непосредственно через анализируемый раствор. Такая схема кондуктометра избавлена от дополнительных погрешностей и наводок, которые характерны для однотрансформаторных приборов. Однако наилучших результатов удается достичь при измерениях, в которых используется активная составляющая математической модели, а информативным параметром, как правило, является ток (или же реактивная составляющая напряжения). Впрочем, и здесь прямыми методами измерений не удается достичь менее, чем 10-процентной погрешности. Но она, в отличие от однотрансформаторных моделей, по крайней мере, линейна и растет по всему диапазону измерений.

Преимущества многообмоточной схемы кондуктометра

Все вышесказанное позволяет сделать вывод, что наилучшие характеристики имеет *многообмоточная схема кондуктометра,* выполняемого в трансформаторном варианте. Она работает с использованием компенсационного метода измерения, характеризуется постоянной чувствительностью по всему диапазону, СКП имеет линейную зависимость от удельной электропроводимости, погрешность не превышает 2 проц. по всему диапазону измерений. Таким образом, проведенные исследования подтвердили, что именно многообмоточная схема кондуктометра, использующая информативный параметр по току, является наилучшей благодаря высокой точности, широкому диапазону измерений и простоте аппаратного оформления.

1.2 Разработка волнового кондуктометра

Обмен протонов между гелем-донором и гелем-акцептором, обеспечивающий сорбцию низкомолекулярных ионов произвольной природы из водных растворов, является сравнительно медленным процессом. Соответственно, для оптимизации систем, использующих эффект Джумадилова, существенным является определение не только кинетических параметров, но и пространственного распределения таких величин как концентрация низкомолекулярных компонент. Обеспечение достаточно высоких скоростей сорбции в промышленных установках по указанным причинам не позволяет работать в условиях полного завершения процессов, приводящих к появлению эффекта Джумадилова.

Сказанное поясняет рисунок 1.1, на котором представлена схема протекания эффекта Джумадилова.

В соответствии с этой схемой, протоны от геля-донора (1) переносятся к гелю-акцептору (2), что, теоретически, должно было бы привести к появлению избыточного электростатического заряда у обоих образцов. Этого, однако, не происходит, что проще всего проследить на примере донорно-акцепторной пары, размещенной в растворе низкомолекулярной соли.

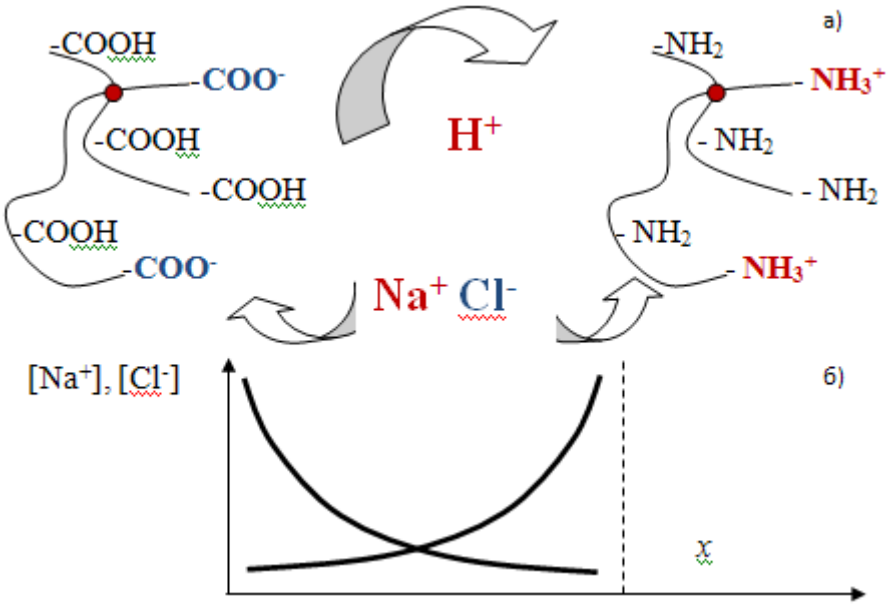


Рисунок 1.1 – Схема протекания эффекта Джумадилова (а) и ожидаемые профили низкомолекулярных ионов (б)

В этом случае подвижные катионы, образующиеся в результате электролитической диссоциации низкомолекулярной компоненты, сорбируются гелем-донором протона, который иначе должен был бы приобрести избыточный отрицательный заряд. Сходным образом, анионы, т.е. отрицательно заряженные частицы, сорбируются гелем-акцептором протонов, который иначе должен был бы приобрести избыточный положительный заряд.

Механизм рассмотренного явления (эффекта Джумадилова) был описан ранее [4], однако, в цитируемых работах не были изучены кинетические характеристики процессов, сопровождающих появление эффекта Джумадилова. Вместе с тем, очевидно, что скорость сорбции низкомолекулярной компоненты будет лимитироваться скоростью переноса протонов от геля-донора к гелю-акцетору. Очевидно также, что сорбция будет протекать параллельно с переносом протона, что должно приводить к появлению неоднородного распределения концентраций в промежутке между гелями, т.к. в первую очередь образцы геля захватят ионы из слоев, непосредственно примыкающих к их поверхности. Можно априори утверждать, что перенос зарядов из толщи раствора к поверхности гидрогелей вызовет неоднородное распределение подвижных частиц.

Следовательно, измеряя профили концентраций сорбируемых низкомолекулярных ионов, можно определить все требуемые кинетические характеристики процессов, сопровождающих эффект Джумадилова.

Для измерения данного профиля в материалах отчета за предыдущий этап был предложен так называемый кондуктометр волнового режима, схема размещения электродов показана на рисунке 1.2.

Разрабатываемый программно-аппаратный комплекс «волнового кондуктометра», таким образом, предназначен для измерения профилей концентраций в растворе. Принцип действия прибора состоит в следующем. В сосуд с раствором вставляется линейка электродов, состоящая из n электродов, равноудаленных друг от друга (рисунок 1.2).

Пары электродов поочередно подключаются к генератору импульсного напряжения и устройству, измеряющему падение напряжение на участке между электродами (рабочий объем). Переключение осуществляется в тактируемом режиме с помощью электронных ключей. В первый такт сигнал с генератора подается на 1-й электрод, съем измеряемого сигнала осуществляется со второго; на следующем такте импульсный сигнал подается уже на 2-й электрод, а снимаются данные с 3-го; на третьем такте подключается к генератору 3-й электрод, а 4-й к съемному устройству, и так далее.

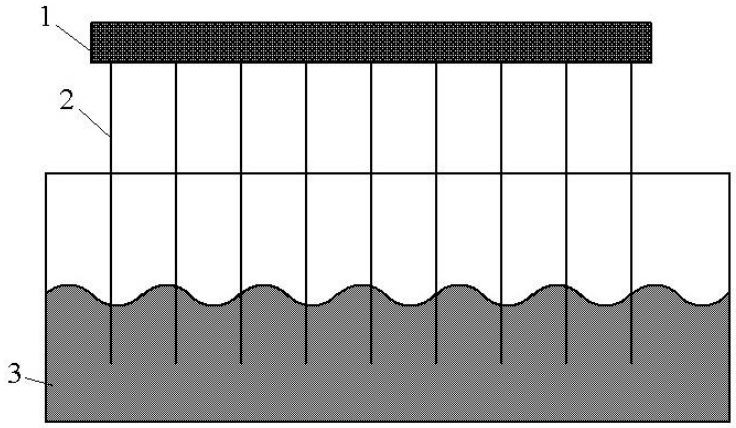


Рисунок 1.2 – Схема расположение электродов: 1 – шина подсоединения электродов, 2 – электроды, 3 – емкость с раствором

В целом, при использовании такой схемы каждый из электродов поочередно служит катодом и анодом, причем в каждый момент времени ток протекает только через одну пару электродов, что иллюстрирует рисунок 1.3.

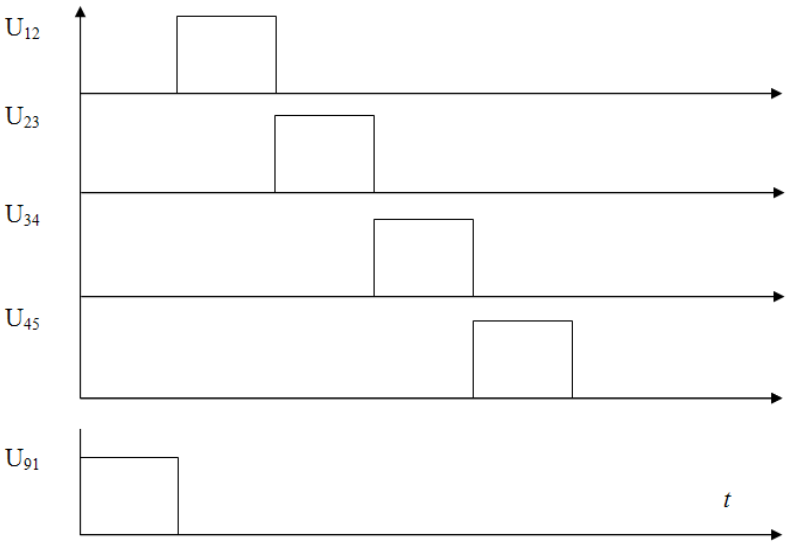


Рисунок 1.3 – Эпюры токов между электродными парами

Такой режим переключения позволяет использовать один прибор для практически одновременного (с точностью до длительности полного цикла переключения между электродными парами) измерения электропроводности в различных точках пространства. Кроме того, поочередное использование электродов в качестве анналов и катодов при повышенной скорости переключения позволяет избежать протекания паразитных электролизных явлений.

Переключение происходит циклически, т.е. как бы в режиме «бегущей» волны. Каждый электрод одинаковое количество времени имеет положительную и отрицательную амплитуду сигнала с генератора, то есть в среднем является электронейтральным, чем исключается какое-либо влияние на раствор и перераспределение концентраций, что позволяет регистрировать пространственное распределение электропроводности при высокой степени синхронизации измерений.

Блок схема аппаратной части показана на рисунке 1.4.

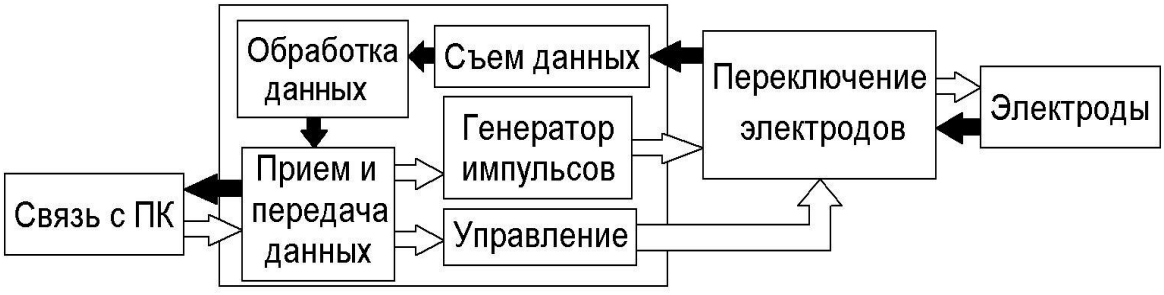


Рисунок 1.4 – Блок схема аппаратной части

Особенностью устройства также является его подключение по USB к компьютеру для сбора и обработки результатов. Волновой кондуктометр имеет модуль приема и передачи данных, который осуществляет эту связь. Программа написана таким образом, что записывает и сохраняет результаты измерения в таблицу, что удобно для обработки и построения графиков распределения концентрации. Оболочка программы построена так, чтобы обеспечить удобство последующей обработки результатов в готовых математических и статистических пакетах.

Программно-аппаратный комплекс проектировался, как это отмечалось в отчете за предыдущий этап, исходя из следующих технических характеристик:

Напряжение питания …….. +5В

Потребляемый ток …….. 40-70 мА

Частота импульсов …….. 250 кГц

Время съема с 9-ти электродов ….. ≈ 0.5 с.

Такие характеристики позволяют использовать стандартные источники питания, а также стандартные комплектующие изделия, работающие в сравнительно низком диапазоне частот.

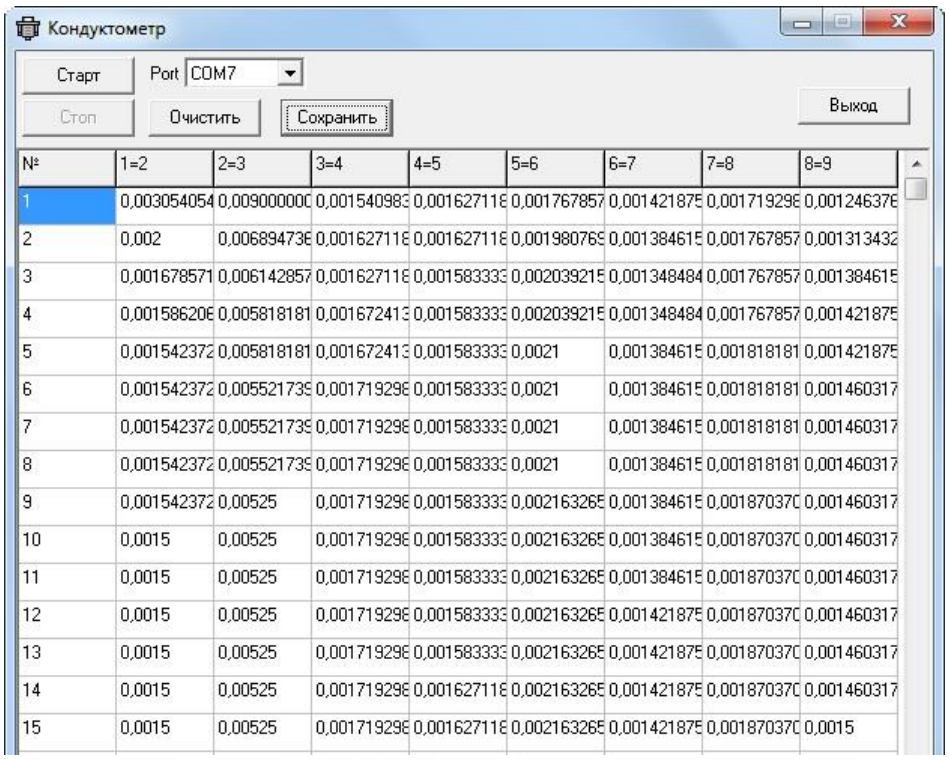
В комплект разрабатываемого устройства изначально входили:

- Плата кондуктометра с переходником COM-USB;
- Блок питания;
- Драйвера и программное обеспечение (Кондуктометр).

В схему изначально входили три основных блока:

- Генератор прямоугольных импульсов, настроенный на частоту 250 кГц (NE555);
- Блок переключения и съема показаний с электродов (ATmega8 + 74HC595);
- Блок связи с компьютером (MAX232).

Указанные блоки остались без изменений, но сами схемы претерпели существенные изменения по сравнению с бета-версией. Диалоговое окно программы осталось прежним (рисунок 1.5). Принципиальная схема кондуктометра волнового режима, удовлетворяющего перечисленным выше требованиям, показана на рисунке 2.5 (альфа-версия).



№	1=2	2=3	3=4	4=5	5=6	6=7	7=8	8=9
1	0,003054054	0,009000000	0,001540983	0,001627118	0,001767857	0,001421875	0,001719298	0,001246376
2	0,002	0,006894736	0,001627118	0,001627118	0,001980769	0,001384615	0,001767857	0,001313432
3	0,001678571	0,006142857	0,001627118	0,001583333	0,002039215	0,001348484	0,001767857	0,001384615
4	0,001586206	0,005818181	0,001672413	0,001583333	0,002039215	0,001348484	0,001767857	0,001421875
5	0,001542372	0,005818181	0,001672413	0,001583333	0,0021	0,001384615	0,001818181	0,001421875
6	0,001542372	0,005521739	0,001719298	0,001583333	0,0021	0,001384615	0,001818181	0,001460317
7	0,001542372	0,005521739	0,001719298	0,001583333	0,0021	0,001384615	0,001818181	0,001460317
8	0,001542372	0,005521739	0,001719298	0,001583333	0,0021	0,001384615	0,001818181	0,001460317
9	0,001542372	0,00525	0,001719298	0,001583333	0,002163265	0,001384615	0,001870370	0,001460317
10	0,0015	0,00525	0,001719298	0,001583333	0,002163265	0,001384615	0,001870370	0,001460317
11	0,0015	0,00525	0,001719298	0,001583333	0,002163265	0,001384615	0,001870370	0,001460317
12	0,0015	0,00525	0,001719298	0,001583333	0,002163265	0,001421875	0,001870370	0,001460317
13	0,0015	0,00525	0,001719298	0,001583333	0,002163265	0,001421875	0,001870370	0,001460317
14	0,0015	0,00525	0,001719298	0,001627118	0,002163265	0,001421875	0,001870370	0,001460317
15	0,0015	0,00525	0,001719298	0,001627118	0,002163265	0,001421875	0,001870370	0,0015

Рисунок 1.5 – Экранная копия оболочки программы

В схему альфа-версии (Приложение Д) внесены следующие изменения, которые устраняют недостатки, выявленные в ходе эксплуатации беты-версии, описанной в материалах отчета за предыдущий этап. Принципиальная схема бета-версии устройства показана на рисунке 1.7 для сравнения. В Приложении Г подробно показан листинг программы, которая и выводит на экран обработанные данные.

Основным отличием альфа-версии от предыдущей является использование кварцевого генератора (элемент Qz1). Он заменяет генератор, собранный на микросхеме NE555.

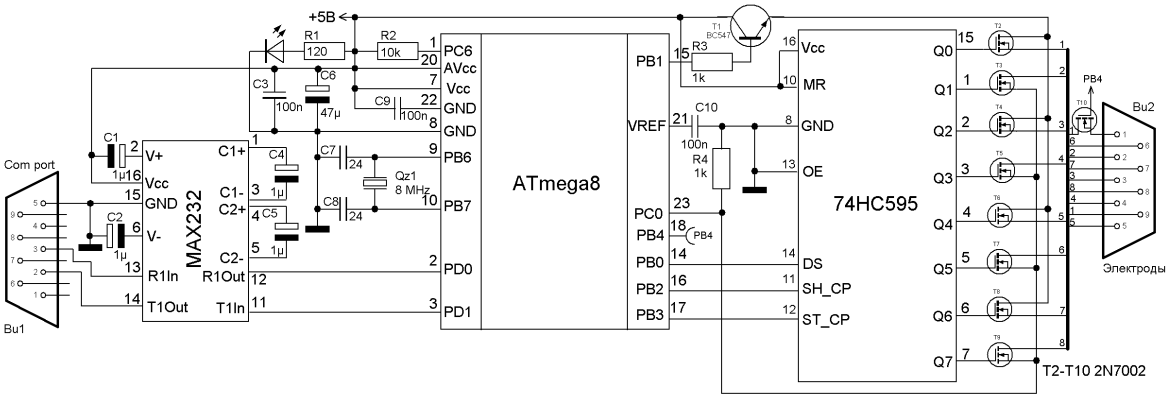


Рисунок 1.6 – Принципиальная схема кондуктометра (альфа-версия)

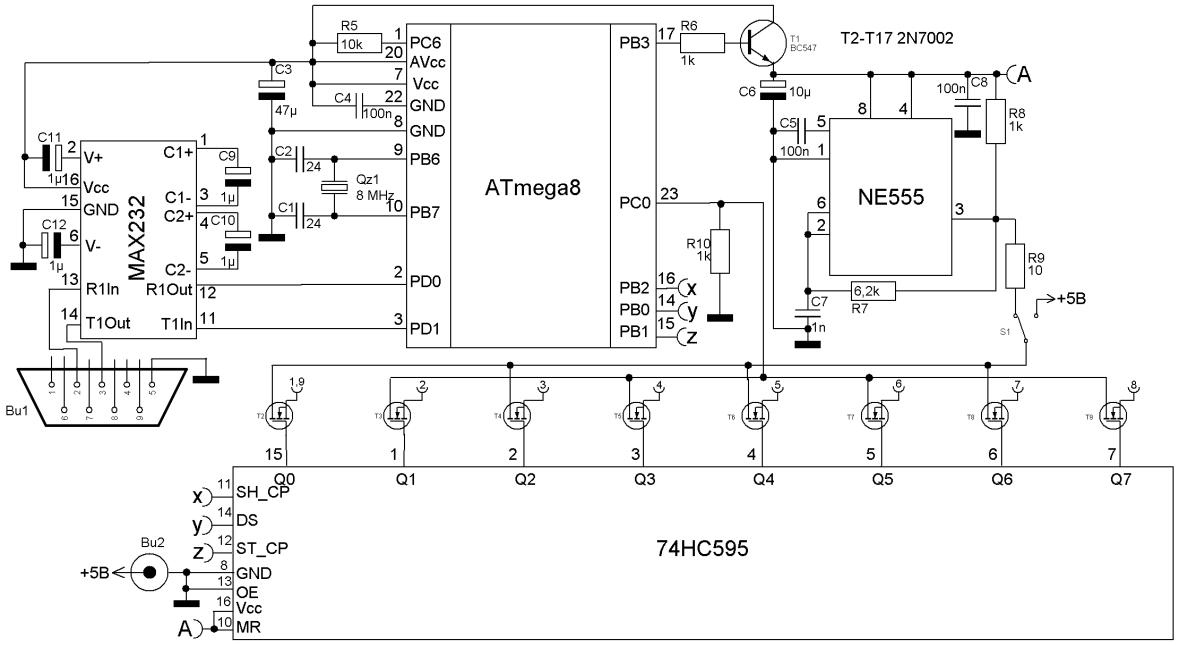


Рисунок 1.7 – Принципиальная схема кондуктометра волнового режима (бета-версия)

Необходимость внесения изменений была продиктована следующим обстоятельствами. Как выяснилось в ходе тестирования бета-версии, генератор на базе был NE555 не в состоянии обеспечить необходимый уровень согласования по времени всех существенных процессов – измерения электропроводности по времени, разбиения всего цикла измерений на такты и времени срабатывания открывающих транзисторных ключей. В результате выдаваемое программой значение электропроводности изменялось от одного измерения к другому, так как при рассогласовании по времени, продолжительность измерений на различных тактах фактически была различной. Данное затруднение удалось преодолеть при переходе к стабилизации при помощи кварцевого генератора.

Остальные блоки схемы работают в том же режиме, что и для случая бета-версии. А именно, при нажатии кнопки «Старт» в программе «Кондуктометр» (описывается ниже), микроконтроллер ATmega8 с помощью сдвигового регистра 74HC595, последовательно подключает электроды ко входу АЦП микроконтроллера. В микроконтроллере имеется десятиразрядное АЦП, с его помощью напряжение на входе преобразуется в число, которое посредством блока связи (MAX232) отправляется в переходник COM-USB, а далее в компьютер. Программа принимает отправленное число, пересчитывает его в проводимость между данными электродами и выводит данные в таблицу, которая может быть преобразована в формат приложения Excel.

Главной частью схемы является микроконтроллер (МК) ATmega8. Он выполняет роль задающего генератора прямоугольных импульсов ворот для всех электродов, снимает данные с них, а также формирует кодовые посылки и отправляет их на ПК.

МК, как и в бета-версии осуществляет последовательный съем данных с электродов. Для этого используется сдвиговый регистр 74HC595. В первый момент времени на его выходах Q0-Q7 выставляется комбинация 11000000, что соответствует открытию транзисторов Т2 и Т3. Прямоугольные импульсы с генератора через транзистор Т2 идут на 1-й электрод, далее, через раствор – на 2-й электрод, и через транзистор Т3 поступают на вход АЦП МК. Данный АЦП является 10-ти разрядным, позволяет измерять изменение напряжения на входе с точностью до 0.004882 В, что заведомо превышает необходимый диапазон точности измерений.

Схема съема представляет собой простой резистивный делитель, где вместо одного сопротивления подсоединяется раствор. Тогда, чтобы посчитать сопротивление раствора, необходимо воспользоваться формулой

$$R_p = \left(\frac{U_{пит}}{U_{АЦП}} - 1\right) * R_{10} \quad (1.1)$$

Где $U_{пит}$ – напряжение питания (5В), $U_{АЦП}$ – напряжение, снятое АЦП.

Чтобы получить проводимость, необходимо взять величину обратную сопротивлению

$$\sigma = \frac{1}{R_p} \quad (1.2)$$

Перерасчет производит программа на ПК. МК же в свою очередь отправляет лишь значение напряжения $U_{АЦП}$. Для этого он формирует кодовую последовательность и отправляет ее на микросхему MAX232 которая преобразует уровни питания с 5В, на необходимые для COM порта. Далее МК отправляет кодовую последовательность в сдвиговый регистр, и он на выходах устанавливает значение 01100000, что соответствует открытому транзистору Т3 и Т4, и процесс описанный выше повторяется. Съем данных происходит циклически до тех пор, пока с ПК не поступит сигнал завершающий съем данных.

Программа на ПК работает следующим образом. При нажатии на «Старт» разрешается открытие порта, и в COM порт отправляется кодовая последовательность, разрешающая работу генератора МК. Далее идет постоянный опрос COM порта, и, если приходит последовательность бит, то она записывается в переменную, это и есть наше значение АЦП, далее это значение преобразуется по вышеуказанным формулам в проводимость и записывается в таблицу. При нажатии кнопки «Стоп» отправляется кодовая последовательность, завершающая работу МК, и COM порт закрывается.

Ниже описывается пользовательская инструкция к разработанному программному обеспечению.

Для первого использования подключить переходник COM-USB к компьютеру.

При подключении будет обнаружено новое устройство, на которое необходимо установить драйвера из папки «CH341SER». После успешной установки драйверов в Диспетчере устройств, во кладке Порты (COM и LPT) появится новое устройство «USB-Serial Controller» (без восклицательного знака) и ему будет присвоен номер виртуального COM порта. Далее следует подключить через разъем COM плату кондуктометра. К разъему OUT подключить электроды. Подключить блок питания. Устройство готово к работе.

Для съема данных запустить программу «Кондуктометр».

В окне Port выбирается номер COM порта, который был присвоен устройству (рисунок 1.8).

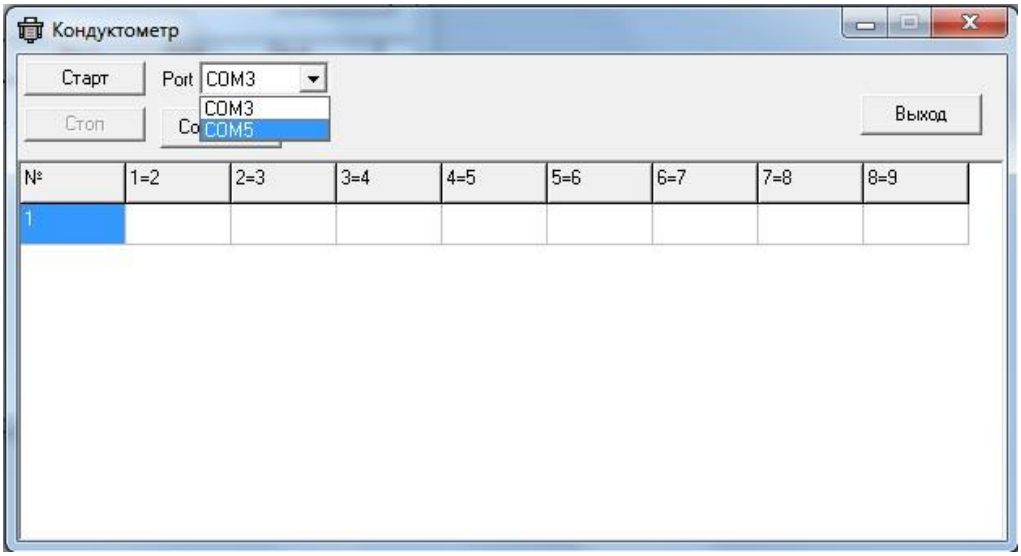


Рисунок 1.8 – Выбор порта в диалоговом окне

Для начала съема данных нажать кнопку «Старт»

При этом таблица начнет заполняться значениями. В колонке 1=2 будет значения проводимости между первым и вторым электродом, 2=3 – вторым и третьим и т.д. Для того чтобы остановить съем данных, необходимо нажать кнопку «Стоп».

Для того чтобы перенести данные таблицы в файл, нажать кнопку «Сохранить», при этом создается файл с расширением .csv, который открывается программой Microsoft Office Excel. Дальнейшую обработку результатов можно проводить в этой программе.

Пример тестирования альфа-версии устройства показан на рисунке 1.10. На данном рисунке показаны тестовые зависимости электропроводности от времени, полученные при измерении электропроводности раствора поваренной соли при концентрации 1,5%. Видно, что измерения всех пар электродов дают стабильно близкие одинаковые значения (за исключением пары, относящейся к первому и последнему электродов линейки). Видно также, что полностью устранен недостаток, присущий бета-версии, т.е. разброс значений при измерении одной парой электродов теперь лежит в пределах погрешности.

Таким образом, по результатам исследований в отчетный период создана альфа-версия рабочего программно-аппаратного комплекса, предназначенного для измерения пространственно-временного распределения электропроводности раствора, обладающего пространственным и временным разрешением, достаточным для изучения процессов, сопровождающих эффект Джумадилова. Тестовые зависимости электропроводности приведены на рисунке 1.10.

Фотография готовой печатной платы в сборке показана на рисунке 1.11.

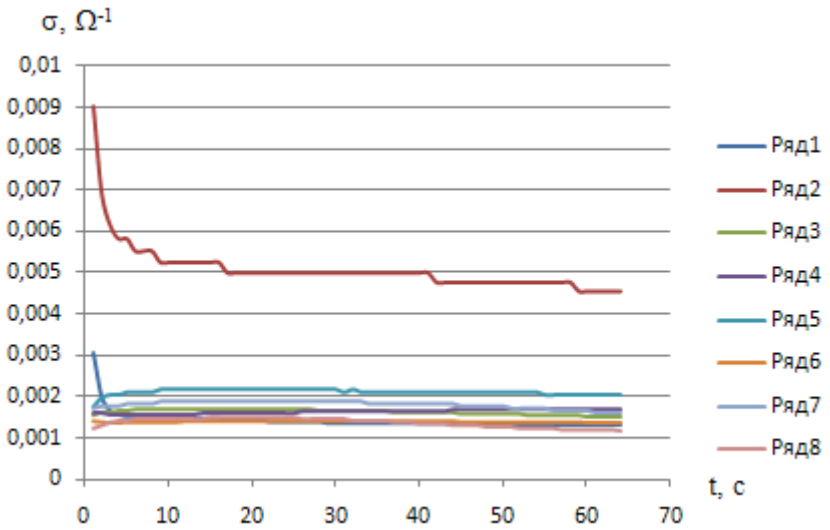


Рисунок 1.10 – Тестовые зависимости электропроводности от времени, полученные на примере 1,5%-ного раствора поваренной соли

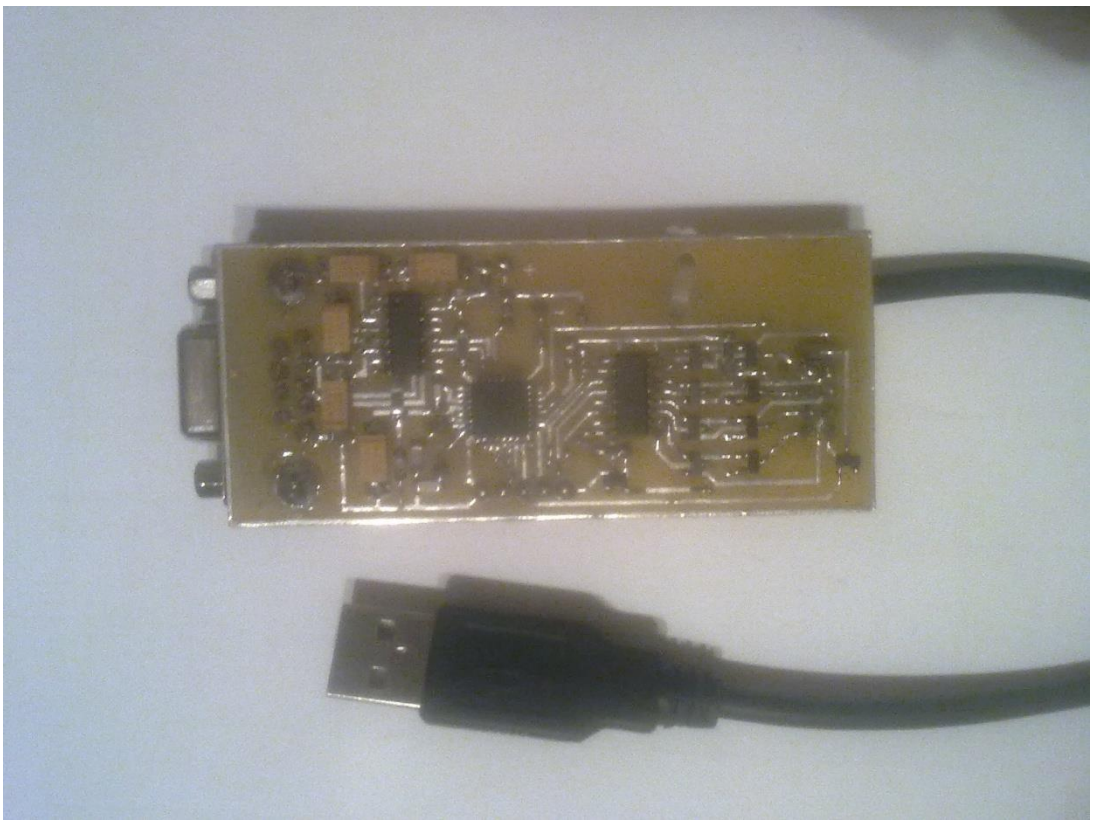


Рисунок 1.11 – Фотография готовой печатной платы программно-аппаратного комплекса, для сравнения масштаба на фотографии представлен USB-разъем, который отвечает за питание.

В таблице указаны перечен элементов для сборки разработанной платы. На рисунке показана печатная плата для стеклолита для разработанного оборудования

Таблица 1.1 - Перечень элементов, использованных для изготовления платы

№	Наименование	Количества
1	Микроконтроллер ATmega8a	1
2	Микросхема MAX232	1
3	Микросхема 74HC595	1
4	Кварц 8 MHz	1
5	Полевой транзистор 2N7002	9
6	Биполярный транзистор BC547	1
7	Разъем COM на плату (мама)	1
8	Электролитические конденсаторы 1 мкФ 16 В 47 мкФ 16 В	 4 1
9	Керамические конденсаторы 100 нФ 24 пФ	 3 2
10	Резисторы	

	120 Ом 1 кОм 10 кОм	1 2 1
11	Перемычка	3
12	Светодиод	1
13	USB шнур	1
14	Штыревые разъемы	-

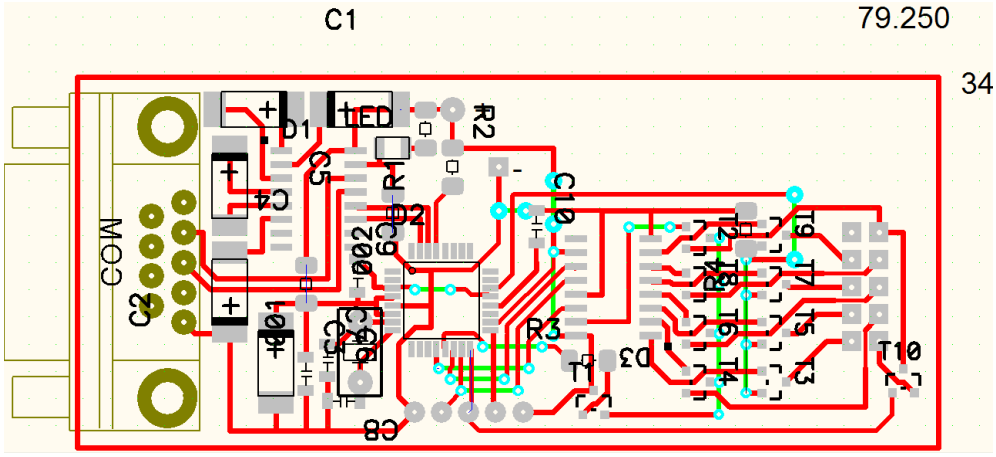


Рисунок 1.12. Печатная плата

Для прошивки микроконтроллера ATmega8 использовала простой программатор для AVR. Программатор работает на основе программы Pony Prog. На рисунке 1.13 показана принципиальная схема, на рисунке 1.14 фотография программатора. Прошивка для микроконтроллера была разработана с помощью AVR Studio 4. Листинг программ приведены в Приложении А и Б.

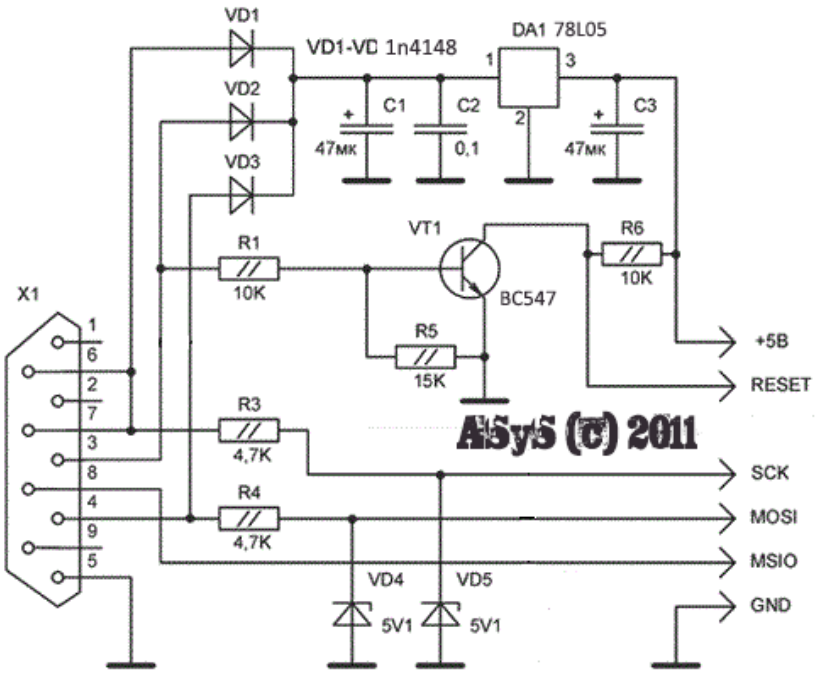


Рисунок 1.13. Принципиальная схема программатора.

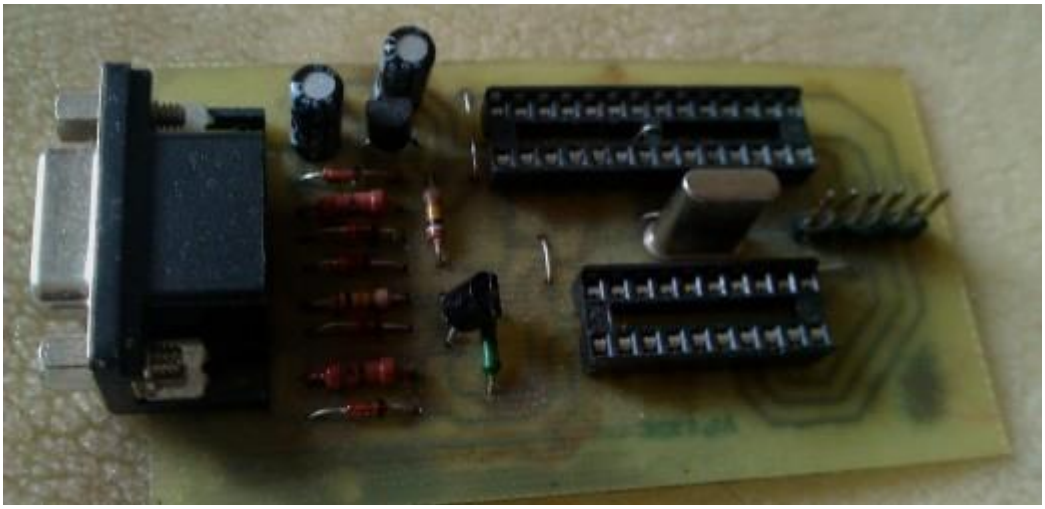


Рисунок 1.14. Фотография программатора.

2 Взаимная активация и высокая селективность полимерных структур в интергелевых системах

2.1 Экспериментальная часть.

Оборудование: Для измерения электропроводности был использован кондуктометр «МАРК 603» (Россия), pH растворов определяли на pH метре “Seven Easy” (METTLER TOLEDO, Китай). Коэффициент набухания $К_Н$ определяли взвешиванием набухших образцов гидрогелей на электронном весе «SHIMADZU AY220» (Япония).

Материалы.

Исследования проводили в среде дистиллированной воды. Гидрогели полиакриловой кислоты были синтезированы в присутствии сшивающего агента N,N-метилен-бис-акриламида и окислительно-востановительной системы $K_2S_2O_8$-$Na_2S_2O_3$. Гидрогель поли-2-метил-5-винилпиридина (гП2М5ВП) был синтезирован из линейного полимера в среде диметилформамида в присуствии эпихдлоргидрина при 60°С.

Синтезированные гидрогели в водной среде составляли интергелевую пару гель поликриловая кислота –гель поли-2-метил-5-винилпиридин (гПАК-гП2М5ВП). Коэффициенты набухания гидрогелей составляли $К_{Н(гПАК)}$=10,1 г/г, $К_{Н(гП2М5ВП)}$=0,46 г/г. Эксперименты проводились при комнатной температуре. Исследования интергелевой системы проводили следующим образом: каждый гидрогель помещался в отдельные специальные стеклянные бюксы, поры которых проницаемы для низкомолекулярных ионов и молекул, но не проходим для дисперсии гидрогелей.

Затем бюксы с гидрогелями помещались в стаканы дистиллированной водой. pH и электропроводность надгелевой жидкости определяли в присутствии гидрогелей в водной среде. Коэффициент набухания вычисляли вычитанием от веса бюкса с гидрогелем и пустого бюкса в соотвествии с формулой:

$$K_{sw} = \frac{m_2 - m_1}{m_1} \qquad (2.1)$$

где m_1 вес сухого гидрогеля, m_2 вес набухшего гидрогеля

2.2 Обоснование результата.

Экспериментальные данные по электропроводности растворов интергелевой системы в зависимости от исходных мольных соотношений и времени представлены на рисунке 2.1. Рисунок отражает изменение электрохимических свойств гидрогелей, составляющих интергелевую систему.

Высокие значения электропроводности в точке максимума указывают на высокие концентрации носителей зарядов. В нашем случае это могут быть

ионы H^+ в водной среде, концентрация которых определяется степенью диссоциации карбоксильных групп полиакриловой кислоты и возрастает в процессе набухания. Однако, в системе присутствует полиоснование гидрогель поли-2-метил-5-винилпиридина, которое может легко присоединить H^+ ионы и перейти в заряженное состояние. Этот процесс должен привести к снижению концентрации ионизованных частиц в растворе.

Появления максимума элетропроводности при соотношении гидрогелей 5:1 можно объяснить низкой концентрацией атомов азота в растворе, так как концентрация винилпиридина в 5 раз меньше, чем концентрация карбоксильных групп. Уменьшение электропроводности с ростом содержания винилпиридин-групп в растворе объясняется увеличением атомов азота ассоцирующихся с протоном.

При эквимольном соотношении гПАК:гП2М5ВП электропроводность достигает минимума, что обуслено максимальной протонизацией звеньев винилпиридина.

В правой части кривой в зависимости χ - гПАК:гП2М5ВП можно было бы ожидать снижения электропроводности за счет связывания H^+ C≥NN группами. Однако, мы наблюдаем рост электропроводности в первой части кривой. На рисунке 3 высокие значения электропроводности которой достигатся при соотношении в точке максимума 1:5, указывает на наибольшее количество ионизованных частиц в растворе.

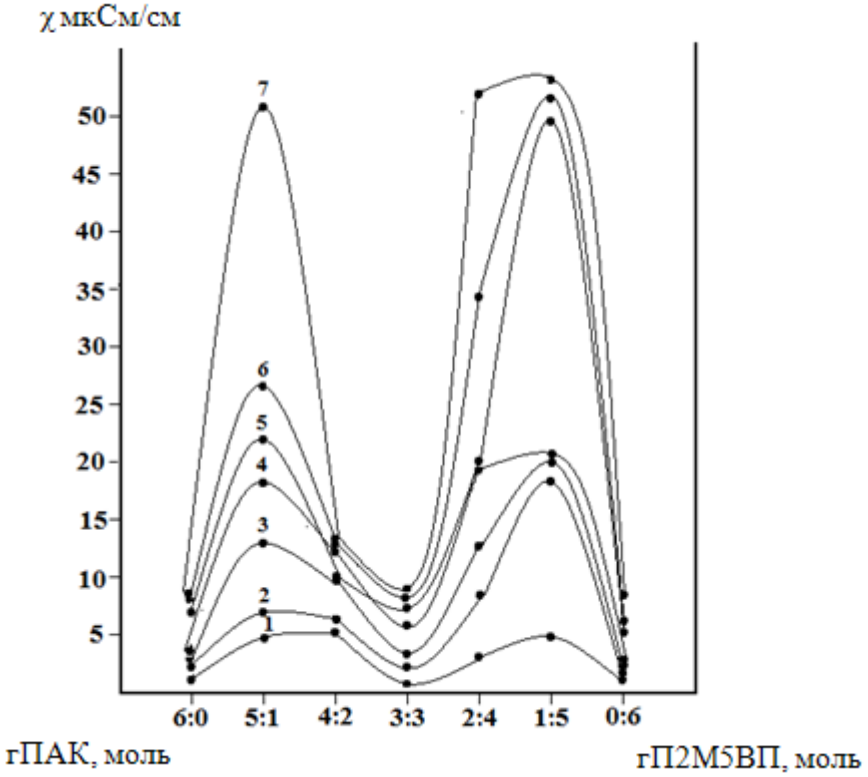


Рисунок 2.1 – Кривые изменения удельной электропроводности системы гидрогель полиакриловой кислоты - гидрогель поли-2-метил-5-пиридина в

зависимости от времени. Описание кривых:1 – 0 мин; 2 – 20 мин; 3 – 40 мин; 4 – 60 мин; 5 – 120 мин; 6 – 240 мин; 7 – 360 мин

Для проверки работоспособности платы был проведен следующий эксперимент: в 180 мл растворе натрий хлорида с концентрацией 0,225% расположили полиакрилат натрия (Sodium polyacrylate) 1 гр. (Рисунок 2.2)

Рисунок 2.2. Наглядный пример ксперимента

Полиакрилат натрия (Sodium polyacrylate) - натриевая соль полимера акриловой кислоты [-CH2-CH(COONa)-]n.

Рисунок 2.3. Частицы полиакрилат натрия

В процессе набухания геля кондуктометр снимает данные фиксирую их на компьютере, которые показаны в таблице 1.

Т а б л и ц а 2.1. Результаты определения проводимости раствора.

№	1=2	2=3	3=4	4=5	5=6	6=7	7=8	8=9
1	0,002333	0,000563	0,001039	0,000722	0,000938	0,000867	0,000914	0,002298
2	0,001885	0,000546	0,000845	0,000685	0,000782	0,000824	0,000824	0,002229

3	0,00183	0,0005	0,000824	0,000685	0,000761	0,000824	0,000824	0,002229
4	0,00183	0,000376	0,000824	0,000685	0,000761	0,000802	0,000824	0,002229
5	0,00183	0,000316	0,000824	0,000685	0,000761	0,000802	0,000824	0,002229
6	0,00183	0,000316	0,000824	0,000685	0,000742	0,000802	0,000824	0,002229
7	0,00183	0,000304	0,000802	0,000685	0,000742	0,000802	0,000824	0,002229
8	0,00183	0,000316	0,000802	0,000685	0,000761	0,000782	0,000824	0,002229
9	0,00183	0,000316	0,000782	0,000685	0,000742	0,000782	0,000824	0,002163
10	0,00183	0,000316	0,000782	0,000685	0,000761	0,000782	0,000824	0,002229
11	0,00183	0,000316	0,000782	0,000685	0,000742	0,000782	0,000824	0,002229
12	0,00183	0,000316	0,000782	0,000685	0,000761	0,000782	0,000824	0,002229
13	0,00183	0,000316	0,000782	0,000685	0,000742	0,000782	0,000824	0,002229
14	0,00183	0,000327	0,000782	0,000685	0,000761	0,000782	0,000845	0,002229
15	0,00183	0,000316	0,000782	0,000667	0,000761	0,000782	0,000824	0,002229
16	0,00183	0,000316	0,000761	0,000685	0,000761	0,000782	0,000845	0,002298
17	0,00183	0,000316	0,000761	0,000667	0,000761	0,000782	0,000845	0,002298
18	0,00183	0,000316	0,000761	0,000685	0,000782	0,000782	0,000845	0,002298
19	0,001885	0,000316	0,000782	0,000667	0,000782	0,000761	0,000845	0,00237
20	0,001885	0,000327	0,000761	0,000667	0,000761	0,000761	0,000845	0,00237
21	0,001885	0,000327	0,000782	0,000667	0,000782	0,000761	0,000845	0,00237
22	0,001885	0,000316	0,000782	0,000667	0,000782	0,000782	0,000867	0,00237
23	0,001941	0,000327	0,000802	0,000649	0,000782	0,000782	0,000867	0,00237
24	0,001885	0,000327	0,000782	0,000632	0,000782	0,000722	0,000867	0,00237
25	0,001885	0,000327	0,000782	0,000598	0,000782	0,000722	0,000867	0,00237
26	0,001885	0,000327	0,000782	0,000615	0,000802	0,000782	0,000867	0,00237
27	0,001941	0,000327	0,000802	0,000598	0,000802	0,000742	0,000824	0,00237
28	0,001885	0,000327	0,000802	0,000598	0,000802	0,000761	0,000867	0,002444
29	0,001941	0,000327	0,000761	0,000582	0,000802	0,000742	0,000867	0,002444
30	0,001941	0,000327	0,000802	0,000598	0,000802	0,000761	0,000867	0,002444
31	0,001941	0,000327	0,000782	0,000598	0,000802	0,000742	0,000867	0,002444
32	0,001941	0,000327	0,000802	0,000598	0,000802	0,000742	0,000867	0,002444
33	0,001941	0,000327	0,000802	0,000582	0,000802	0,000742	0,000867	0,002444
34	0,001941	0,000327	0,000802	0,000582	0,000802	0,000742	0,000867	0,002444
35	0,001941	0,000327	0,000802	0,000598	0,000802	0,000742	0,000867	0,002444
36	0,001941	0,000327	0,000802	0,000582	0,000802	0,000742	0,000867	0,002444
37	0,001941	0,000327	0,000802	0,000582	0,000824	0,000742	0,000867	0,00237
38	0,001941	0,000339	0,000802	0,000598	0,000824	0,000742	0,000867	0,002444
39	0,001941	0,000327	0,000782	0,000582	0,000824	0,000742	0,000845	0,002444
40	0,001941	0,000327	0,000802	0,000582	0,000824	0,000742	0,000867	0,002444
41	0,002	0,000327	0,000802	0,000582	0,000824	0,000742	0,000867	0,002444
42	0,001941	0,000316	0,000802	0,000582	0,000824	0,000742	0,000867	0,002444
43	0,002	0,000327	0,000802	0,000582	0,000824	0,000742	0,000867	0,00237
44	0,002	0,000339	0,000802	0,000582	0,000824	0,000722	0,000867	0,002444
45	0,002	0,000327	0,000802	0,000582	0,000802	0,000703	0,000867	0,00237
46	0,001941	0,000316	0,000802	0,000582	0,000824	0,000742	0,000867	0,002444
47	0,002	0,000327	0,000802	0,000615	0,000824	0,000782	0,000867	0,002605
48	0,002	0,000327	0,000802	0,000598	0,000824	0,000703	0,000867	0,002444

49	0,002	0,000327	0,000802	0,000598	0,000802	0,000722	0,000867	0,002444
50	0,002	0,000327	0,000802	0,000598	0,000824	0,000722	0,000867	0,002444
51	0,002	0,000327	0,000802	0,000615	0,000824	0,000742	0,000867	0,002444
52	0,002	0,000327	0,000802	0,000615	0,000824	0,000761	0,000867	0,002605
53	0,002	0,000339	0,000802	0,000615	0,000824	0,000761	0,00089	0,002523
54	0,002	0,000339	0,000824	0,000615	0,000845	0,000742	0,00089	0,002444
55	0,002	0,000339	0,000824	0,000582	0,000824	0,000703	0,00089	0,002605
56	0,001941	0,000316	0,000802	0,000598	0,000802	0,000722	0,000867	0,002444
57	0,002	0,000327	0,000824	0,000685	0,000845	0,000824	0,00089	0,002605
58	0,002061	0,000351	0,000824	0,000685	0,000867	0,000824	0,00089	0,00269
59	0,002	0,000339	0,000845	0,000685	0,000824	0,000761	0,000867	0,002523
60	0,002	0,000316	0,000824	0,000615	0,000824	0,000703	0,00089	0,002523
61	0,001941	0,000327	0,000802	0,000632	0,000824	0,000761	0,000867	0,002605
62	0,002	0,000339	0,000824	0,000722	0,000845	0,000824	0,00089	0,00269
63	0,002061	0,000351	0,000824	0,000703	0,000845	0,000782	0,00089	0,002605
64	0,002061	0,000339	0,000824	0,000703	0,000845	0,000782	0,00089	0,002523
65	0,002061	0,000339	0,000824	0,000685	0,000845	0,000742	0,000867	0,002444
66	0,002	0,000316	0,000782	0,000649	0,000845	0,000761	0,00089	0,002605
67	0,002	0,000327	0,000824	0,000703	0,000824	0,000782	0,00089	0,00278
68	0,002061	0,000351	0,000845	0,000742	0,000867	0,000824	0,00089	0,002523
69	0,002061	0,000351	0,000824	0,000722	0,000867	0,000802	0,00089	0,002605
70	0,002061	0,000339	0,000824	0,000703	0,000845	0,000802	0,000914	0,002605
71	0,002061	0,000351	0,000824	0,000667	0,000845	0,000761	0,00089	0,002523
72	0,002	0,000339	0,000824	0,000632	0,000845	0,000761	0,000914	0,002605
73	0,001941	0,000327	0,000802	0,000615	0,000824	0,000761	0,00089	0,00278
74	0,002061	0,000339	0,000845	0,000742	0,000867	0,000802	0,00089	0,00269
75	0,002061	0,000351	0,000845	0,000722	0,000867	0,000802	0,00089	0,00269
76	0,002	0,000339	0,000824	0,000667	0,000845	0,000782	0,000914	0,00269
77	0,002	0,000339	0,000824	0,000667	0,000845	0,000761	0,00089	0,002605
78	0,002	0,000327	0,000824	0,000632	0,000824	0,000761	0,00089	0,002605
79	0,002	0,000327	0,000802	0,000632	0,000824	0,000761	0,00089	0,002605
80	0,002	0,000339	0,000824	0,000667	0,000845	0,000782	0,00089	0,002605
81	0,002	0,000327	0,000782	0,000615	0,000824	0,000761	0,000914	0,002875
82	0,002	0,000339	0,000824	0,000649	0,000845	0,000802	0,000914	0,002875
83	0,002061	0,000364	0,000845	0,000703	0,000867	0,000802	0,00089	0,00269
84	0,002061	0,000351	0,000845	0,000667	0,000867	0,000782	0,00089	0,002605
85	0,002	0,000351	0,000824	0,000632	0,000845	0,000742	0,000867	0,002605
86	0,002	0,000316	0,000802	0,000667	0,000845	0,000761	0,00089	0,00269
87	0,002	0,000327	0,000824	0,000742	0,000845	0,000824	0,000914	0,002974
88	0,002	0,000339	0,000824	0,000703	0,000867	0,000802	0,00089	0,00278
89	0,002061	0,000351	0,000824	0,000703	0,000867	0,000802	0,000914	0,00269
90	0,002	0,000339	0,000824	0,000667	0,000845	0,000782	0,00089	0,00269
91	0,002	0,000339	0,000824	0,000685	0,000845	0,000802	0,00089	0,002605
92	0,002	0,000339	0,000824	0,000649	0,000845	0,000782	0,00089	0,00269
93	0,002	0,000327	0,000802	0,000632	0,000824	0,000782	0,00089	0,002875
94	0,002	0,000351	0,000824	0,000722	0,000867	0,000845	0,000938	0,002875

95	0,002061	0,000364	0,000845	0,000703	0,000867	0,000802	0,000914	0,00278
96	0,002	0,000327	0,000802	0,000632	0,000824	0,000761	0,000845	0,00269
97	0,001941	0,000293	0,000782	0,000649	0,000824	0,000824	0,000914	0,002875
98	0,002	0,000351	0,000824	0,000761	0,000867	0,00089	0,000938	0,002974
99	0,002061	0,000339	0,000824	0,000703	0,000867	0,000824	0,000914	0,00278
100	0,002	0,000327	0,000824	0,000667	0,000845	0,000782	0,000914	0,00278

Изменение проводимости в растворе можно увидеть на графике 1.

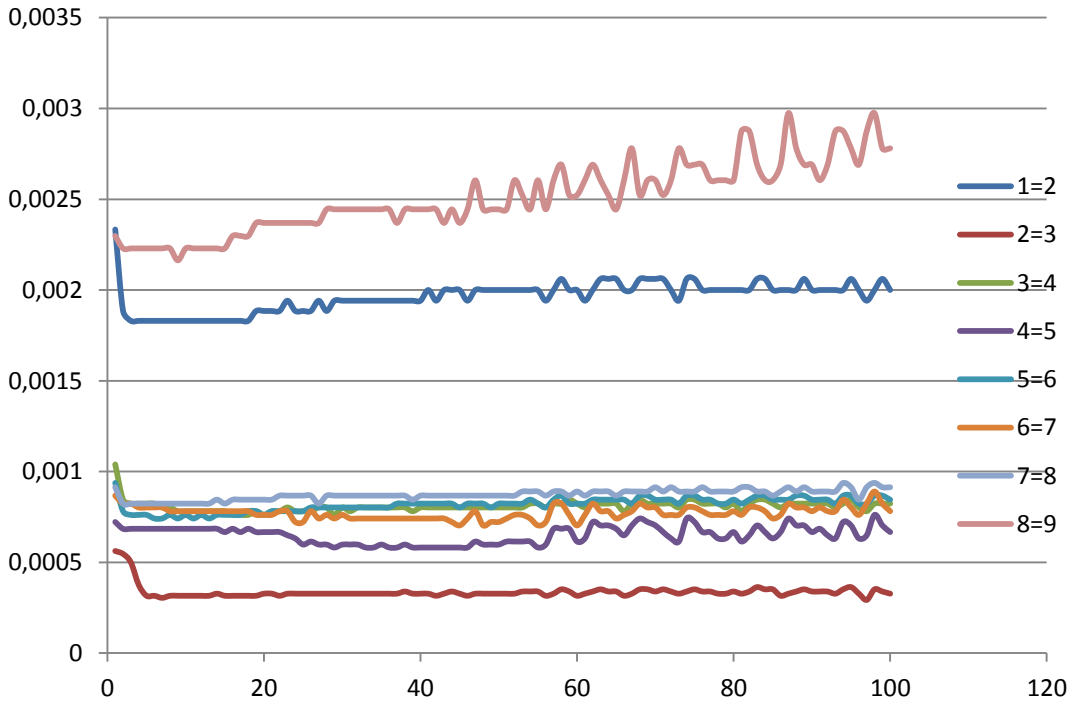


Рисунок 2.3. График изменения проводимости раствора

3 Расчетное обоснование методики проведения эксперимента

Поведение низкомолекулярных ионов в среде между образцами гидрогелей описывается в самом общем случае следующими уравнениями?

$$\frac{\partial n_+}{\partial t} = D_+ \nabla^2 n_+ - \nabla\left(b_+ eEn_+\right) \tag{3.1}$$

$$\frac{\partial n_-}{\partial t} = D_- \nabla^2 n_- + \nabla\left(b_- eEn_-\right) \tag{3.2}$$

$$\nabla E = 4\pi e\left(n_+ - n_- - N_0\right) \tag{3.3}$$

где (n_+, n_-) - концентрации положительных и отрицательных заряженных частиц, E - напряженность электростатического поля, N_0 - заряд сетки (в данной записи предполагается, что он является отрицательным), $D_\pm, b_\pm$ - коэффициента диффузии и подвижности, соответственно, e - элементарный заряд.

Приведенные выше соображения позволяют рассматривать передачу ионов от геля-донора к гелю-акцептору в квазистационарном режиме. Это означает, что в уравнениях (1) и (2) можно пренебречь производными по времени и переписать систему уравнений (1) – (3) в следующем виде

$$\frac{dn_+}{dx} - \frac{eE}{kT} n_+ = -\frac{j_+}{D_+} \tag{3.4}$$

$$\frac{dn_-}{dx} + \frac{eE}{kT} n_- = \frac{j_-}{D_-} \tag{3.5}$$

$$\frac{d}{dx} E = 4\pi e\left(n_+ - n_- - N_0\right) \tag{3.6}$$

где предполагается, что параметры, характеризующие систему, зависят только от одной пространственной переменной, j_+, j_- - константы интегрирования. Такое предположение адекватно отражает характер исследуемого процесса при условии, что образцы геля представляют собой плоские пластины, расположенные параллельно друг другу, а расстояние между ними существенно меньше, чем поперечные размеры образцов.

Чтобы определить константы интегрирования j_+, j_-, рассмотрим область, удаленную от гидрогелей (вспомогательных электродов и т.д.). В такой области с высокой точностью выполняется условие нейтральности среды

$$n_{+0} = n_{-0}, \tag{3.7}$$

которое означает также, в данной области электрическое поле постоянно (однородно). Соответственно, в данной области все производные по пространственными переменным равны нулю, что сразу позволяет записать условие

$$\frac{j_+}{D_+} = \frac{j_-}{D_-} \tag{3.8}$$

Используем (3.8) и вычтем уравнения (3.4) и (3.5) друг из друга. Имеем

$$\frac{d}{dx}(n_+ + n_-) - \frac{eE}{kT}(n_+ - n_-) = 0 \tag{3.9}$$

Выражая разность концентраций через электрическое поле при помощи (3.6), получаем

$$\frac{d}{dx}(n_+ + n_-) - \frac{eE}{kT}\left(\frac{1}{4\pi}\frac{dE}{dx} - N_0\right) = 0 \tag{3.10}$$

Сходным образом, складывая уравнения (3.4) и (3.5), имеем

$$\frac{d}{dx}(n_+ - n_-) - \frac{eE}{kT}(n_+ + n_-) = -\frac{j_+}{D_+} - \frac{j_-}{D_-} \tag{3.11}$$

откуда

$$\frac{d}{dx}\left(\frac{1}{4\pi e}\frac{dE}{dx}\right) - \frac{eE}{kT}(n_+ + n_-) = -\frac{j_+}{D_+} - \frac{j_-}{D_-} \tag{3.12}$$

Уравнения (3.10) и (3.12) представляют собой замкнутую систему на две неизвестных функции – суммарную концентрацию и электрическое поле. Данная система имеет третий порядок, но его можно существенно понизить.

Более того, в рассматриваемом случае процедура понижения порядка уравнений позволяет получить вариационную форму уравнений движения ионов, что является принципиальным для математического описания эффекта Джумадилова в динамическом режиме.

Как отмечалось выше, лимитирующей по времени стадией является перенос ионов через раствор сравнительно низкой концентрации, обладающий низкой электропроводностью. Поэтому соответствующие функционалы необходимо получить именно для этой области. Необходимо подчеркнуть, что рассматриваемая система не является равновесной, поэтому для ее описания использовать стандартные термодинамические функции, вообще говоря, не является корректным. Поэтому аналоги соответствующих термодинамических потенциалов должны быть получены независимым способом, например, непосредственно на основе анализа уравнений движения ионов.

Для области свободного раствора записанное выше уравнение (3.10) принимает вид

$$\frac{d}{dx}\left(n_+ + n_-\right) - \frac{1}{8\pi kT}\frac{dE^2}{dx} = 0 \tag{3.13}$$

Такая форма записи позволяет немедленно получить первый интеграл исследуемой системы дифференциальных уравнений, выражающий некоторое условие сохранения:

$$n_+ + n_- - \frac{E^2}{8\pi kT} = Z \tag{3.14}$$

Отметим, что по форме данное условие сохранения полностью совпадает с полученным ранее в работе [1], однако, условие (14) справедливо, в том числе, для неравновесных систем, которые в цитированной работе не рассматривалось.

Подставляя (14) в (12), получаем

$$\frac{1}{4\pi e}\frac{d^2E}{dx^2} - \frac{eE}{kT}\left(\frac{E^2}{8\pi kT} + Z\right) = -\frac{j_+}{D_+} - \frac{j_-}{D_-} = -\alpha \tag{3.15}$$

или

$$\frac{1}{4\pi e}\frac{d^2E}{dx^2} - \frac{eE^3}{8\pi\left(kT\right)^2} - Z\frac{eE}{kT} + \alpha = 0 \tag{3.16}$$

Умножая уравнение (16) на первую производную электрического поля по координате, после несложных выкладок получаем

$$\frac{1}{8\pi e}\frac{d}{dx}\left(\frac{dE}{dx}\right)^2 - \frac{d}{dx}\frac{eE^4}{32\pi\left(kT\right)^2} - Z\frac{d}{dx}\frac{eE^2}{4kT} + \alpha\frac{dE}{dx} = 0 \tag{3.17}$$

Такая запись позволяет получить следующий интеграл рассматриваемой системы дифференциальных уравнений, который выражает еще одно условие сохранения.

$$\frac{1}{8\pi e}\left(\frac{dE}{dx}\right)^2 - \frac{eE^4}{32\pi\left(kT\right)^2} - Z\frac{eE^2}{4kT} + \alpha E = const = Z_2 \tag{3.18}$$

А именно, на основании (3.18) можно сразу записать такое условие сохранения как

$$2\pi e\left(n_+ - n_-\right)^2 - \frac{eE^4}{32\pi\left(kT\right)^2} - Z\frac{eE^2}{4kT} + \left(\frac{j_+}{D_+} + \frac{j_-}{D_-}\right)E = const = Z_2 \tag{3.19}$$

Значение константы α можно вычислить на основе тех же соображений, что и равенство (8), т.е. рассматривая такую область раствора, в которой поле является постоянным. Имеем

$$\alpha = \frac{j_+}{D_+} + \frac{j_-}{D_-} = \frac{eE_0}{kT}\left(n_{+0} + n_{-0}\right) = 2\frac{eE_0}{kT}c_0 \tag{3.20}$$

Аналогичным образом, константа Z выражается как

$$Z = n_{+0} + n_{-0} - \frac{E_0^2}{8\pi kT} = 2c_0 - \frac{E_0^2}{8\pi kT} \tag{3.21}$$

Подставляя соотношения (20) и (21) в (19), получаем единственное уравнение на поле, развивающееся в системе (существенно, что данное уравнение уже имеет первый порядок)

$$\frac{1}{8\pi e}\left(\frac{dE}{dx}\right)^2 - \frac{eE^4}{32\pi\left(kT\right)^2} - \left(2c_0 - \frac{E_0^2}{8\pi kT}\right)\frac{eE^2}{4kT} + 2\frac{eE_0}{kT}c_0E = const = Z_2 \tag{3.22}$$

Снова рассматривая область однородности электрического поля, можно отыскать значение полученной величины как

$$Z_2 = -\frac{eE_0^4}{32\pi\left(kT\right)^2} - \left(2c_0 - \frac{E_0^2}{8\pi kT}\right)\frac{eE_0^2}{4kT} + 2\frac{eE_0}{kT}c_0E_0 \tag{3.23}$$

или

$$Z_2 = 1{,}5c_0\frac{eE_0^2}{kT} \tag{3.24}$$

Подставляя найденные величины в (22), получаем

$$\frac{1}{8\pi e^2}\left(\frac{dE}{dx}\right)^2 - \frac{E^4}{32\pi\left(kT\right)^2} - \left(2c_0 - \frac{E_0^2}{8\pi kT}\right)\frac{E^2}{4kT} + 2\frac{E_0}{kT}c_0E - 1{,}5c_0\frac{E_0^2}{kT} = 0 \tag{3.25}$$

Вид уравнения (25) позволяет перейти к безразмерной искомой функции

$$f^2 = \frac{E^2}{c_0kT} \tag{3.26}$$

Разделив (3.25) на величину e, получаем

$$\frac{kT}{8\pi e^2 c_0}\left(\frac{df}{dx}\right)^2 - \frac{f^4}{32\pi} - \left(2 - \frac{f_0^2}{8\pi}\right)\frac{f^2}{4} + 2ff_0 - \frac{3}{2}f_0^2 = 0 \qquad (3.27)$$

Как и следовало ожидать, в уравнение (3.27) в качестве параметра входит длина Дебая

$$\lambda^2 = \frac{kT}{8\pi e^2 c_0} \qquad (3.28)$$

Таким образом, итоговое уравнение зависит только от длины Дебая λ и приведенного квадрата электрического поля

$$f_0^2 = \frac{E_0^2}{c_0 kT} \qquad (3.29)$$

В безразмерных величинах полученное уравнение может быть переписано в форме, удобной для решения численными методами

$$\lambda^2\left(\frac{df}{dx}\right)^2 - \frac{f^4}{32\pi} - \left(2 - \frac{f_0^2}{8\pi}\right)\frac{f^2}{4} + 2ff_0 - \frac{3}{2}f_0^2 = 0, \qquad (3.30)$$

Причем если расстояния измерять в единицах длины Дебая, то полученное уравнение зависит только от одного параметра

$$\left(\frac{df}{dx}\right)^2 - \frac{f^4}{32\pi} - \left(2 - \frac{f_0^2}{8\pi}\right)\frac{f^2}{4} + 2ff_0 - \frac{3}{2}f_0^2 = 0, \qquad (3.31)$$

На рис. 3.1 представлено семейство кривых, отвечающих правым частям уравнения (3.31),

$$S = \frac{f^4}{32\pi} + \left(2 - \frac{f_0^2}{8\pi}\right)\frac{f^2}{4} - 2ff_0 + \frac{3}{2}f_0^2, \qquad (3.31)$$

Использовались следующее значение управляющего параметра (3.29)

$f(0) =$ 7,14,21, 28, 35

Видно, что при определенных значениях управляющего параметра рассматриваемая величина приобретает значения, меньшие нуля. Это означает, что область определения переменной в правой части уравнения (3.31) существенно ограничена.

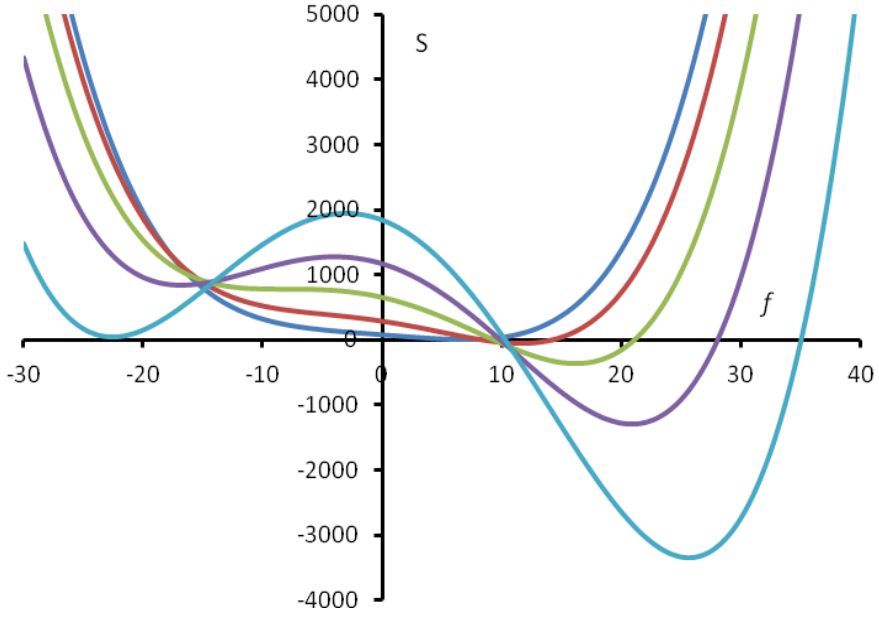


Рисунок 3.1.

Интерес представляют ветви, которые лежат в области значений f, больших f_0, причем непосредственно из рис.3.1 видно, что представляющие интерес решения при выборе отрицательного знака корня асимптотически сходятся к f_0.

Вернемся к уравнению (3.16), которое, с учетом соотношений (3.20) и (3.21) может быть переписано как

$$\frac{1}{4\pi e}\frac{d^2E}{dx^2}-\frac{eE^3}{8\pi(kT)^2}-\left(2c_0-\frac{E_0^2}{8\pi kT}\right)\frac{eE}{kT}+2\frac{eE_0}{kT}c_0=0 \qquad (3.32)$$

или

$$\frac{1}{4\pi e}\frac{d^2E}{dx^2}-\frac{eE\left(E^2-E_0^2\right)}{8\pi(kT)^2}-2c_0\frac{e\left(E-E_0\right)}{kT}=0 \qquad (3.33)$$

Как известно, отыскать экстремум функционала приведенного ниже вида, зависящего от функции и ее производной

$$\int_a^b\left(\frac{df}{dx}\right)^2+F\left(f\left(x\right)\right)dx \qquad (3.34)$$

можно, составляя дифференциальное уравнение

$$2\frac{d}{dx}\left(\frac{df}{dx}\right)-\frac{\partial F}{\partial f}=0 \qquad (3.35)$$

Сравнивая (35) и (32) можно видеть, что задача об отыскании решения (32) эквивалентна задаче об отыскании экстремума следующего функционала.

$$\frac{1}{4\pi e}\int_a^b\left(\frac{dE}{dx}\right)^2+\frac{eE^4}{32\pi(kT)^2}+\left(2c_0-\frac{E_0^2}{8\pi kT}\right)\frac{eE^2}{2kT}-2\frac{eE_0E}{kT}c_0\,dx \qquad (3.36)$$

Полученный результат, во-первых, представляет академический интерес, так как впервые показано, что система, заведомо удаленная от состояния равновесия, также может описываться на языке вариационных принципов, причем соответствующие функционалы являются прямым следствием уравнения движения частиц.

Во-вторых, полученный функционал имеет также и практическое значение. Действительно, профиль электрического поля при проведении инженерных расчетов далеко не всегда нужно знать с высокой точностью. Для практических нужд его можно, например, аппроксимировать набором отрезков прямых, что позволяет отыскать приближенный профиль, просто решая систему алгебраических уравнений. Кроме того, именно вариационная форма наиболее удобна для решения различного рода зада оптимизации. И, наконец, именно вариационная форма вида (3.36) позволяет наиболее естественным образом учесть граничное условие, вытекающее из характера постановки экспериментов. Действительно, в экспериментах чаще всего известно не граничное значение электрического поля, а напряжение, приложенное к соответствующему промежутку. В используемых обозначениях такое граничное условие записывается как

$$\int_a^b\frac{df}{dx}dx=C=const \qquad (3.37)$$

Следовательно, используя метод неопределенных множителей Лагранжа, полный функционал, обеспечивающий решение поставленной задачи с учетом естественного граничного условия, может быть записан в следующей форме.

$$\frac{1}{4\pi e}\int_a^b\left(\frac{dE}{dx}\right)^2+\frac{eE^4}{32\pi(kT)^2}+\left(2c_0-\frac{E_0^2}{8\pi kT}\right)\frac{eE^2}{2kT}-2\frac{eE_0E}{kT}c_0\,dx+\mu\int_a^b\frac{df}{dx}dx, \qquad (3.38)$$

где μ - неопределенный множитель Лагранжа.

Функционал вида (38), таким образом, является основой для разрабатываемой методики инженерного расчета, так как этот результат достаточен для построения соответствующего программного обеспечения, основанного на методах линейного программирования.

Заключение

Разработаны модификации принципиальных радиоэлектронных схемы программно-аппаратного комплекса, обеспечивающего проведение измерений пространственно-временного распределения электропроводности раствора, осуществлено их макетирование и выполнено сопутствующее программное обеспечение в альфа-версии. Проведено тестирование готового изделия, доказана его пригодность для изучения динамических процессов, сопутствующих эффекту Джумадилова.

Проведены предварительные эксперименты, доказывающие возможность существенного повышения скорости протекания эффекта Джумадилова за счет самопроизвольного возникновения дополнительных электрических полей в интергелевых системах при протекании потока жидкости.

Доказана необходимость создания специального радиоэлектронного оборудования, обеспечивающего регистрацию пространственных профилей распределения концентрации.

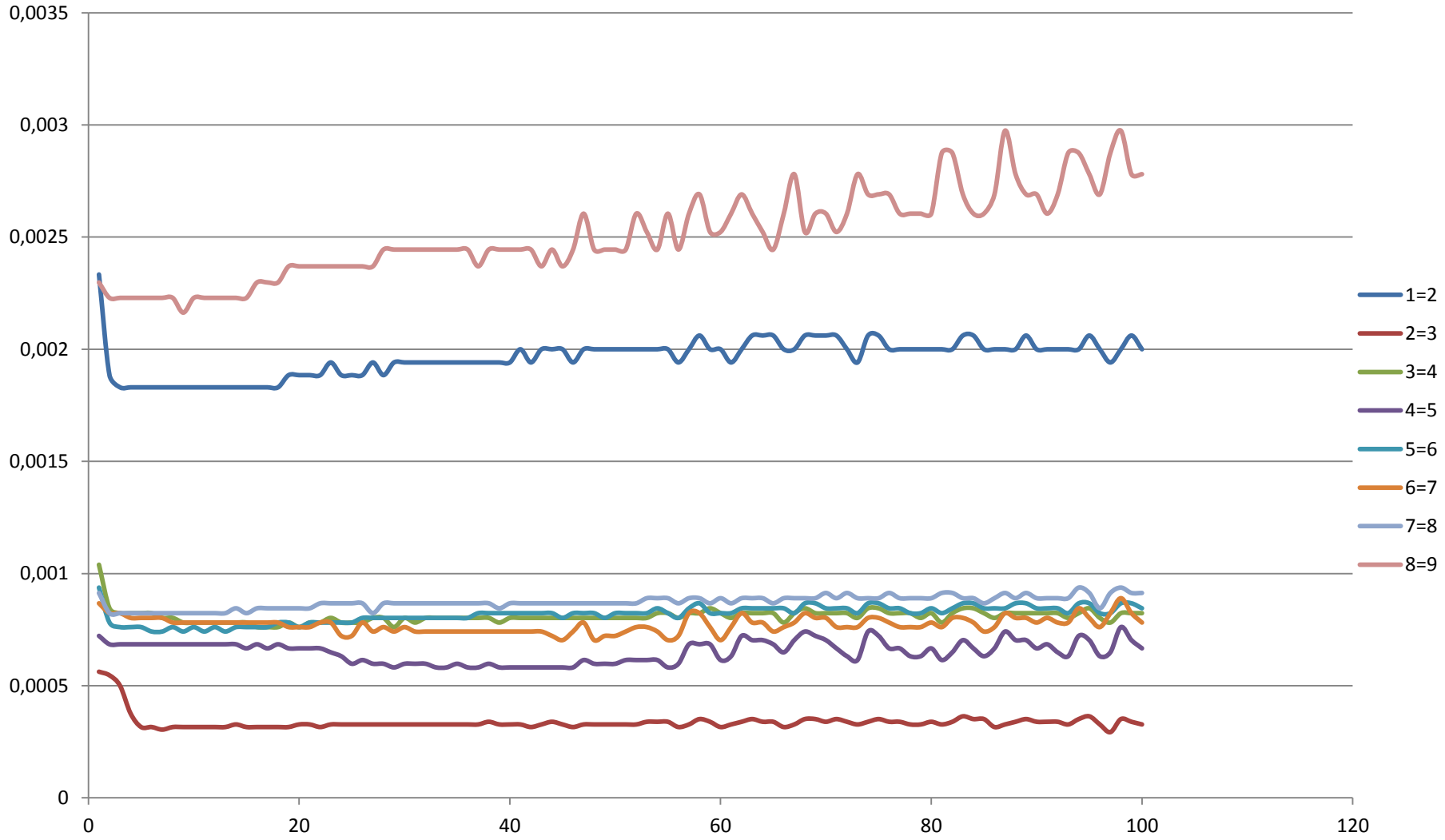
Проведена разработка и осуществлено предварительное макетирование соответствующего радиоэлектронного оборудования; выполнено сопутствующее программное обеспечение и проведено его тестирование в бета-версии.

Дальнейшее исследование проявлений эффекта Джумадилова требует создания нового оборудования, обеспечивающего измерение профилей электропроводности в динамическом режиме с достаточно высоким пространственным и временным разрешением. В данном отчете описана модификация волнового кондуктометра (альфа-версия), обеспечивающая проведение измерений такого рода и доказана его работоспособность на тестовых измерениях.

Список использованной литературы

1. Бектуров Е.А., Джумадилов Т.К. Новые подходы к изучению эффекта дистанционного взаимовлияния гидрогелей. Изв. НАН РК, 2009,№1, С. 86-87.
2. Бектуров Е.А., Джумадилов Т.К., Корганбаева Ж.К.. Дистанционный эффект в полимерных системах. Вестник КазНУ, сер.хим. 2010, №3(59). С. 108-110.
3. Джумадилов Т.К. Эффект дистанционного взаимодействия полимерных гидрогелей в инновационной технологии. Промышленность Казахстана, 2011, №2, С.70-72.
4. Jumadilov T., Shaltykova D., Suleimenov I. Anomalous ion exchange phenomenon // Austrian-slovenian polymer meeting, ASPM 2013- Slovenia, 3-5apr. 2013 - Abstr.S5-P51.
5. Jumadilov T.K. Mutual activation and high selectivity of polymeric structures in intergel systems. Abstr. 3rd International Caucasian Symposium on Polymers and Advanced Materials, *Tbilisi, Georgia..P. 48.*
6. Budtova T. V., Suleimenov I. E., Frenkel S. A diffusion approach to description of swelling of polyelectrolyte hydrogels //Polymer science. – 1995. – Т. 37. – №. 1. – С. 10-16.
7. http://www.ngpedia.ru/id101765p1.html - Большая Энциклопедия Нефти Газа.
8. Белов Л. В. Самоучитель разработчика устройств на микроконтроллерах AVR. «Наука и техника», Санк-Петербург, 2008.
9. Белов А.В. Микроконтроллеры в радиолюбительской практике. «Наука и техника», Санк-Петербург, 2007.
10. Брайан Керниган, Деннис Ритчи. Язык программирования С. «Невский диалект», Санк-Петербург, 2001.
11. Справочник химика. -2-е изд., перераб. - М.: Химия, 1987.- т. 1-4.

Приложение А. Изменение проводимости раствора



Приложение Б. Листинг прошивки микроконтроллера на AVR Studio 4

```
#define F_CPU 8000000UL

#include <avr/io.h>
#include <avr/interrupt.h>
#include <util/delay.h>
#include <string.h>
#include <stdio.h>

int b;
int n=8;
unsigned int u=0; //Глобальная переменная с содержимым преобразования

 //Функция инициализации модуля USART микроконтроллера AtMega8
    void USART_Init( unsigned int ubrr)
    {
    UBRRH = (unsigned char)( ubrr>>8 );//Декларируем переменную для скорости
соединения 9600
    UBRRL = (unsigned char)ubrr;

    UCSRB = 1<<RXEN|1<<TXEN|1<<RXCIE|0<<TXCIE; // Вкл передатчик,
приемник, прерывания по приходу
//    UCSRC = 0<<UMSEL|1<<UCSZ0|1<<UCSZ1; // 8-bit
         UCSRC = (1<<URSEL)|(0<<UCSZ2)|(1<<UCSZ1)|(1<<UCSZ0); // 8-bit
    }

// Прием данных с ПК
unsigned char uart_getc( void )
{
   //ждем приема байта
   while( ( UCSRA & ( 1 << RXC ) ) == 0  );
   //считываем принятый байт
   return UDR;
}

 // Передача данных в ПК
void uart_putc( char c )
{
  //ждем окончания передачи предыдущего байта
  while( ( UCSRA & ( 1 << UDRE ) ) == 0 );
  UDR = c;
}
```

```
 // Передача текста
void uart_puts( char *str )
{
  unsigned char c;
  while( ( c = *str++ ) != 0 ) {
    uart_putc( c );
  }
}


unsigned int getADC(void) //Считывание АЦП
{ unsigned int v;

ADCSRA|=(1<<ADSC);//Начать преобразование

while ((ADCSRA&_BV(ADIF))==0x00); //Дождатся окончания преобразования
 //ADCL = ADCL-0x90;
 //ADCH = ADCH-0x01;
//v=(ADCL|ADCH<<8);
v=ADCH;
return v;
}

ISR(TIMER2_OVF_vect)      // Сдивиговый регист по прерываниям
{
_delay_ms(3);

PORTB &= ~_BV(PB3);          // St=0
b++;

if (b % 2 == 0) PORTB |= _BV(PB1); else PORTB &= ~_BV(PB1); // ШИМ 333 Гц

if (n==8) PORTB |= _BV(PB4); else PORTB &= ~_BV(PB4);

if (n == 1) {
          if (b == 1) PORTB |= _BV(PB0);
               else
                    if (b == 8) PORTB |= _BV(PB0);
                   else PORTB &= ~_BV(PB0);
          PORTB |= _BV(PB2); // sp=1
          PORTB &= ~_BV(PB2);  //sp=0
               }
else {
      if (n == b) {
```

```
                PORTB |= _BV(PB0);
                PORTB |= _BV(PB2); // sp=1
        PORTB &= ~_BV(PB2);  //sp=0
                PORTB |= _BV(PB0);
        }

                else PORTB &= ~_BV(PB0);
        PORTB |= _BV(PB2); // sp=1
        PORTB &= ~_BV(PB2);  //sp=0
}

if (b==8) {n--; PORTB |= _BV(PB3);  // St=1 //защелкиваем введенные данные
                        b=0;
                                                        u=getADC();
//uart_putc((u/256)&0x0003); uart_putc(u&0x00ff);
                                                        uart_putc(u);
                                                        }

if (n==0) {n=8; b=0;}

TCNT2 = (256-12);
}

char c;

int main( void )
{
PORTD |= 0x03; // RXD и TXD подтягиваются в +U

  DDRD &= ~0x01; // RXD вход
  DDRD |= 0x02;  // TXD выход

DDRB = 0b00011111;      // PB0,PB1,PB2,PB3,PB4 - выход

USART_Init (51);//Инициализируем модуль USART

        // Настройка таймера
TCCR2 = (0<<CS22)|(0<<CS21)|(1<<CS20);   // без делителя
TIMSK = 0x40;                       //По переполнению
TCNT2 = (256-12);

// Настройка АЦП
ADCSRA=(1<<ADEN)|(1<<ADPS2)|(1<<ADPS1)|(0<<ADPS0);  //Включаем
АЦП, тактовая частота преобразователя СК/64 от тактовой микроконтроллера
```

```
ADMUX=(0<<REFS1)|(1<<REFS0)|(1<<ADLAR)|(1<<MUX0)|(1<<MUX1)|(1<<
MUX2)|(0<<MUX3); // сравнение АЦП с AVcc, смещение битов влево (для
чтения 8 бит), выствление ADC7


  while( 1 ) {

    char c = uart_getc();            // Принимаем с ПК "1"
    if (c == '1') {//PORTB |= _BV(PB3);  // Выдаем 1 в PB3
                      sei();}            // Разрешаем прерывания
          else
          {//PORTB &= ~_BV(PB3);;
           cli(); } // Выдаем 0 в PB3, запрещаем прерывания

    // uart_putc( c );  // Отправляем в ПК данные

  }
  return 0;
}
```

Приложение В. Листинг в 16-разряде

```
:1000000012C02CC02BC02AC048C028C027C026C0A0
:1000100025C024C023C022C021C020C01FC01EC0D4
:100020001DC01CC01BC011241FBECFE5D4E0DEBF25
:10003000CDBF10E0A0E6B0E0EAEBF1E002C0059031
:100040000D92A236B107D9F710E0A2E6B0E001C0E8
:100050001D92A736B107E1F78ED0ADC0D1CF90BDCC
:1000600089B988E98AB986E880BD08955F9BFECF8B
:100070008CB108955D9BFECF8CB90895FC0103C03F
:100080005D9BFECF8CB981918823D1F70895369A74
:10009000349BFECF85B190E008951F920F920FB66A
:1000A0000F9211242F933F938F939F9380E797E1B3
:1000B0000197F1F7C39880916400909165000196D3
:1000C000909365008093640080FD02C0C19A01C0D6
:1000D000C19820916000309161002830310511F401
:1000E000C49A0BC0C4982130310539F48130910590
:1000F00051F08830910549F406C02817390729F4D2
:10010000C09AC29AC298C09A01C0C098C29AC298B6
:100110000897B1F42150304030936100209360008 3
:10012000C39A109265001092640036 9A349BFECFF9
:1001300085B18093620010926300 5D9BFECF8CB905
:10014000809160009091610089 2B51F488E090E0EB
:100150009093610080936000109265001092640 09B
:1001600084EF84BD9F918F913F912F910F900FBE8F
:100170000F901F90189582B3836082BB8898899AEC
:100180008FE187BB10BC83E389B988E98AB996E817
:1001900090BD81E085BD80E489BF84EF84BD96B9C0
:1001A00087E687B95F9BFECF8CB1813311F47894D9
:0A01B000F9CFF894F7CFF894FFCFD1
:0201BA0008003B
:00000001FF
```

Приложение Г. Листинг программы на Delphi 7

```
program Terminal;

uses
  Forms,
  Main in 'Main.pas' {MainForm};

{$R *.RES}

begin
  Application.Initialize;
  Application.Title := 'Кондуктометр';
  Application.CreateForm(TMainForm, MainForm);
  Application.Run;
end.
----------------------------------------------------------------------------------------------------
unit Main;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  StdCtrls, ExtCtrls, BCPort, Grids;

type
  TMainForm = class(TForm)
    Panel1: TPanel;
    cbPort: TComboBox;
    Label1: TLabel;
    btnConnect: TButton;
    btnDisconnect: TButton;
    BComPort1: TBComPort;
    StringGrid1: TStringGrid;
    Button1: TButton;
    Button2: TButton;
    Button3: TButton;
    procedure FormCreate(Sender: TObject);
    procedure btnConnectClick(Sender: TObject);
    procedure btnDisconnectClick(Sender: TObject);
    procedure BComPort1RxChar(Sender: TObject; Count: Integer);
    procedure Button1Click(Sender: TObject);
    procedure Button2Click(Sender: TObject);
    procedure Button3Click(Sender: TObject);
```

```
  private

  end;

var
  MainForm: TMainForm;
  i,j:integer;
implementation

{$R *.DFM}
{$R Led.res}

function StrToHex(source: String): String;
var i:integer;
    c:Char;
    s:String;
begin
    s := '';
    for i:=1 to Length(source) do
    begin
      c := source[i];
      s := s +  IntToHex(Integer(c),2);
    end;
    result := s;
end;

procedure TMainForm.FormCreate(Sender: TObject);
begin
  EnumComPorts(cbPort.Items);
  cbPort.ItemIndex := 0;
  j:=1;
  StringGrid1.Cells[0,0]:='№';
  StringGrid1.Cells[0,1]:='1';
  StringGrid1.Cells[1,0]:='1=2';
  StringGrid1.Cells[2,0]:='2=3';
  StringGrid1.Cells[3,0]:='3=4';
  StringGrid1.Cells[4,0]:='4=5';
  StringGrid1.Cells[5,0]:='5=6';
  StringGrid1.Cells[6,0]:='6=7';
  StringGrid1.Cells[7,0]:='7=8';
  StringGrid1.Cells[8,0]:='8=9';
end;

procedure TMainForm.btnConnectClick(Sender: TObject);
```

```
begin
Button3.Enabled:=false;
  BComPort1.Port := cbPort.Text;
  if BComPort1.Open then
  begin
    btnConnect.Enabled := False;
    cbPort.Enabled := False;
    btnDisconnect.Enabled := True;
    BComPort1.WriteStr('1');
  end;
end;

procedure TMainForm.btnDisconnectClick(Sender: TObject);
begin
  BComPort1.WriteStr('2');
  if BComPort1.Close then
  begin
    btnConnect.Enabled := True;
    Button3.Enabled:=true;
    cbPort.Enabled := True;
    btnDisconnect.Enabled := False;
  end;
end;

procedure TMainForm.BComPort1RxChar(Sender: TObject; Count: Integer);
var
  S: String; k:integer; R: real;
begin
  BComPort1.ReadStr(S, Count);
  i:=i+1;
  if i=9 then begin
  StringGrid1.RowCount:=j+2;
  i:=1; j:=j+1;
  StringGrid1.Cells[0,j]:=IntToStr(j); end;
  S:=StrToHex(S);     // преобразование в шеснадцатиричное число
  S:='$'+S;
  k:=StrToInt(S);     // Преобразование hex to dec
  if k=0 then R:=0 else begin
  R:=k*0.02;          // Напряжение на АЦП
  if (i=1) or (i=2) then
  R:=(3/R-1)*1000
  else R:=(3.1/R-1)*1000; // Сопротивление раствора
  R:=1/R;           // Проводимость
  end;
```

```
  StringGrid1.Cells[i,j]:=FloatToStr(R);

end;

procedure TMainForm.Button1Click(Sender: TObject);
begin
close;
end;

procedure TMainForm.Button2Click(Sender: TObject);
var List: TStringList;  u,k: Integer;  s,b:string;
begin
 List:=TStringList.Create;
with StringGrid1 do
  for u:=0 to RowCount-1 do begin s:='';
  for k:=0 to  ColCount-1 do
    s:=s+Cells[k,u]+';';
    List.Add(s);
  end;

b:='Result.csv';
List.SaveToFile(b);

end;

procedure TMainForm.Button3Click(Sender: TObject);
begin
StringGrid1.Rows[1].Clear;
StringGrid1.Cells[0,1]:='1';
i:=0; j:=1;
StringGrid1.RowCount:=2;
end;

end.
```

```
unit BCPort;

interface

uses
  Windows, Messages, SysUtils, Classes;

{$B-,H+,X+}
```

```
{$IFDEF VER140}
  {$DEFINE D6UP}
{$ENDIF}

{$IFDEF VER150}
  {$DEFINE D6UP}
{$ENDIF}

type
  TBaudRate = (br110, br300, br600, br1200, br2400, br4800, br9600, br14400,
    br19200, br38400, br56000, br57600, br115200, br128000, br256000);
  TByteSize = (bs5, bs6, bs7, bs8);
  TComErrors = set of (ceFrame, ceRxParity, ceOverrun, ceBreak, ceIO, ceMode,
                       ceRxOver, ceTxFull);
  TComEvents = set of (evRxChar, evTxEmpty, evRing, evCTS, evDSR, evRLSD,
                       evError, evRx80Full);
  TComSignals = set of (csCTS, csDSR, csRing, csRLSD);
  TParity = (paNone, paOdd, paEven, paMark, paSpace);
  TStopBits = (sb1, sb1_5, sb2);
  TSyncMethod = (smThreadSync, smWindowSync, smNone);
  TComSignalEvent = procedure(Sender: TObject; State: Boolean) of object;
  TComErrorEvent = procedure(Sender: TObject; Errors: TComErrors) of object;
  TRxCharEvent = procedure(Sender: TObject; Count: Integer) of object;

  TOperationKind = (okWrite, okRead);
  TAsync = record
    Overlapped: TOverlapped;
    Kind: TOperationKind;
    Data: Pointer;
    Size: Integer;
  end;
  PAsync = ^TAsync;

  TBComPort = class;

  TComThread = class(TThread)
  private
    FComPort: TBComPort;
    FEvents: TComEvents;
    FStopEvent: THandle;
  protected
    procedure DoEvents;
    procedure Execute; override;
    procedure SendEvents;
```

```
  procedure Stop;
public
  constructor Create(AComPort: TBComPort);
  destructor Destroy; override;
end;

TComTimeouts = class(TPersistent)
private
  FComPort: TBComPort;
  FReadInterval: Integer;
  FReadTotalM: Integer;
  FReadTotalC: Integer;
  FWriteTotalM: Integer;
  FWriteTotalC: Integer;
  procedure SetComPort(const AComPort: TBComPort);
  procedure SetReadInterval(const Value: Integer);
  procedure SetReadTotalM(const Value: Integer);
  procedure SetReadTotalC(const Value: Integer);
  procedure SetWriteTotalM(const Value: Integer);
  procedure SetWriteTotalC(const Value: Integer);
protected
  procedure AssignTo(Dest: TPersistent); override;
public
  constructor Create;
  property ComPort: TBComPort read FComPort;
published
  property ReadInterval: Integer read FReadInterval write SetReadInterval;
  property ReadTotalMultiplier: Integer read FReadTotalM write SetReadTotalM;
  property ReadTotalConstant: Integer read FReadTotalC write SetReadTotalC;
  property WriteTotalMultiplier: Integer
    read FWriteTotalM write SetWriteTotalM;
  property WriteTotalConstant: Integer
    read FWriteTotalC write SetWriteTotalC;
end;

TBComPort = class(TComponent)
private
  FBaudRate: TBaudRate;
  FByteSize: TByteSize;
  FConnected: Boolean;
  FCTPriority: TThreadPriority;
  FEvents: TComEvents;
  FEventThread: TComThread;
  FHandle: THandle;
```

```
  FInBufSize: Integer;
  FOutBufSize: Integer;
  FParity: TParity;
  FPort: String;
  FStopBits: TStopBits;
  FSyncMethod: TSyncMethod;
  FTimeouts: TComTimeouts;
  FUpdate: Boolean;
  FWindow: THandle;
  FOnCTSChange: TComSignalEvent;
  FOnDSRChange: TComSignalEvent;
  FOnError: TComErrorEvent;
  FOnRing: TNotifyEvent;
  FOnRLSDChange: TComSignalEvent;
  FOnRx80Full: TNotifyEvent;
  FOnRxChar: TRxCharEvent;
  FOnTxEmpty: TNotifyEvent;
  procedure CallCTSChange;
  procedure CallDSRChange;
  procedure CallError;
  procedure CallRing;
  procedure CallRLSDChange;
  procedure CallRx80Full;
  procedure CallRxChar;
  procedure CallTxEmpty;
  procedure SetBaudRate(const Value: TBaudRate);
  procedure SetByteSize(const Value: TByteSize);
  procedure SetCTPriority(const Value: TThreadPriority);
  procedure SetInBufSize(const Value: Integer);
  procedure SetOutBufSize(const Value: Integer);
  procedure SetParity(const Value: TParity);
  procedure SetPort(const Value: String);
  procedure SetStopBits(const Value: TStopBits);
  procedure SetSyncMethod(const Value: TSyncMethod);
  procedure SetTimeouts(const Value: TComTimeouts);
  procedure WindowMethod(var Message: TMessage);
protected
  procedure ApplyBuffer;
  procedure ApplyDCB;
  procedure ApplyTimeouts;
  procedure CreateHandle;
  procedure DestroyHandle;
  procedure SetupComPort;
public
```

```
    constructor Create(AOwner: TComponent); override;
    destructor Destroy; override;
    procedure AbortAllAsync;
    procedure BeginUpdate;
    procedure ClearBuffer(Input, Output: Boolean);
    function Close: Boolean;
    procedure EndUpdate;
    function InBufCount: Integer;
    function IsAsyncCompleted(AsyncPtr: PAsync): Boolean;
    function Open: Boolean;
    function OutBufCount: Integer;
    function Read(var Buffer; Count: Integer): Integer;
    function ReadAsync(var Buffer; Count: Integer; var AsyncPtr: PAsync): Integer;
    function ReadStr(var Str: string; Count: Integer): Integer;
    function ReadStrAsync(var Str: string; Count: Integer; var AsyncPtr: PAsync):
Integer;
    procedure SetDTR(State: Boolean);
    procedure SetRTS(State: Boolean);
    function Signals: TComSignals;
    function WaitForAsync(var AsyncPtr: PAsync): Integer;
    function Write(const Buffer; Count: Integer): Integer;
    function WriteAsync(const Buffer; Count: Integer; var AsyncPtr: PAsync):
Integer;
    function WriteStr(const Str: string): Integer;
    function WriteStrAsync(const Str: string; var AsyncPtr: PAsync): Integer;
    property Connected: Boolean read FConnected;
    property CTPriority: TThreadPriority read FCTPriority write SetCTPriority;
  published
    property BaudRate: TBaudRate read FBaudRate write SetBaudRate;
    property ByteSize: TByteSize read FByteSize write SetByteSize;
    property InBufSize: Integer read FInBufSize write SetInBufSize;
    property OutBufSize: Integer read FOutBufSize write SetOutBufSize;
    property Parity: TParity read FParity write SetParity;
    property Port: String read FPort write SetPort;
    property SyncMethod: TSyncMethod read FSyncMethod write SetSyncMethod;
    property StopBits: TStopBits read FStopBits write SetStopBits;
    property Timeouts: TComTimeouts read FTimeouts write SetTimeouts;
    property OnCTSChange: TComSignalEvent read FOnCTSChange write
FOnCTSChange;
    property OnDSRChange: TComSignalEvent read FOnDSRChange write
FOnDSRChange;
    property OnError: TComErrorEvent read FOnError write FOnError;
    property OnRing: TNotifyEvent read FOnRing write FOnRing;
```

```
    property OnRLSDChange: TComSignalEvent read FOnRLSDChange write
FOnRLSDChange;
    property OnRx80Full: TNotifyEvent read FOnRx80Full write FOnRx80Full;
    property OnRxChar: TRxCharEvent read FOnRxChar write FOnRxChar;
    property OnTxEmpty: TNotifyEvent read FOnTxEmpty write FOnTxEmpty;
  end;

  EComPort = class(Exception);

procedure InitAsync(var AsyncPtr: PAsync);
procedure DoneAsync(var AsyncPtr: PAsync);
procedure EnumComPorts(Ports: TStrings);

procedure Register;

implementation

uses
  Forms;

const
  CM_COMPORT = WM_USER + 1;

  CEMess: array[1..15] of string =
    ('Попытка открыть несуществующий COM-порт',
     'Порт занят другой программой',
     'Ошибка записи в порт',
     'Ошибка чтения из порта',
     'Недопустимый параметр Async',
     'Ошибка функции PurgeComm',
     'Не удалось получить статус асинхронной операции',
     'Ошибка функции SetCommState',
     'Ошибка функции SetCommTimeouts',
     'Ошибка функции SetupComm',
     'Ошибка функции ClearCommError',
     'Ошибка функции GetCommModemStatus',
     'Ошибка функции EscapeCommFunction',
     'Нельзя менять свойство, пока порт открыт',
     'Ошибка обращения к системному реестру');

function EventsToInt(const Events: TComEvents): Integer;
begin
  Result := 0;
  if evRxChar in Events then Result := Result or EV_RXCHAR;
```

```
  if evTxEmpty in Events then Result := Result or EV_TXEMPTY;
  if evRing in Events then Result := Result or EV_RING;
  if evCTS in Events then Result := Result or EV_CTS;
  if evDSR in Events then Result := Result or EV_DSR;
  if evRLSD in Events then Result := Result or EV_RLSD;
  if evError in Events then Result := Result or EV_ERR;
  if evRx80Full in Events then Result := Result or EV_RX80FULL;
end;

function IntToEvents(Mask: Integer): TComEvents;
begin
  Result := [];
  if (EV_RXCHAR and Mask) <> 0 then Result := Result + [evRxChar];
  if (EV_TXEMPTY and Mask) <> 0 then Result := Result + [evTxEmpty];
  if (EV_RING and Mask) <> 0 then Result := Result + [evRing];
  if (EV_CTS and Mask) <> 0 then Result := Result + [evCTS];
  if (EV_DSR and Mask) <> 0 then Result := Result + [evDSR];
  if (EV_RLSD and Mask) <> 0 then Result := Result + [evRLSD];
  if (EV_ERR and Mask) <> 0 then Result := Result + [evError];
  if (EV_RX80FULL and Mask) <> 0 then Result := Result + [evRx80Full];
end;

{ TComThread }

constructor TComThread.Create(AComPort: TBComPort);
begin
  inherited Create(True);
  FStopEvent := CreateEvent(nil, True, False, nil);
  FComPort := AComPort;
  Priority := FComPort.CTPriority;
  SetCommMask(FComPort.FHandle, EventsToInt(FComPort.FEvents));
  Resume;
end;

destructor TComThread.Destroy;
begin
  Stop;
  inherited Destroy;
end;

procedure TComThread.Execute;
var
  EventHandles: array[0..1] of THandle;
  Overlapped: TOverlapped;
```

```
  Signaled, BytesTrans, Mask: DWORD;
begin
  FillChar(Overlapped, SizeOf(Overlapped), 0);
  Overlapped.hEvent := CreateEvent(nil, True, True, nil);
  EventHandles[0] := FStopEvent;
  EventHandles[1] := Overlapped.hEvent;
  repeat
    WaitCommEvent(FComPort.FHandle, Mask, @Overlapped);
    Signaled := WaitForMultipleObjects(2, @EventHandles, False, INFINITE);
    if (Signaled = WAIT_OBJECT_0 + 1) and
      GetOverlappedResult(FComPort.FHandle, Overlapped, BytesTrans, False) then
    begin
      FEvents := IntToEvents(Mask);
      case FComPort.SyncMethod of
        smThreadSync: Synchronize(DoEvents);
        smWindowSync: SendEvents;
        smNone      : DoEvents;
      end;
    end;
  until Signaled <> (WAIT_OBJECT_0 + 1);
  SetCommMask(FComPort.FHandle, 0);
  PurgeComm(FComPort.FHandle, PURGE_TXCLEAR or PURGE_RXCLEAR);
  CloseHandle(Overlapped.hEvent);
  CloseHandle(FStopEvent);
end;

procedure TComThread.Stop;
begin
  SetEvent(FStopEvent);
  Sleep(0);
end;

procedure TComThread.SendEvents;
begin
  if evError in FEvents then
    SendMessage(FComPort.FWindow, CM_COMPORT, EV_ERR, 0);
  if evRxChar in FEvents then
    SendMessage(FComPort.FWindow, CM_COMPORT, EV_RXCHAR, 0);
  if evTxEmpty in FEvents then
    SendMessage(FComPort.FWindow, CM_COMPORT, EV_TXEMPTY, 0);
  if evRing in FEvents then
    SendMessage(FComPort.FWindow, CM_COMPORT, EV_RING, 0);
  if evCTS in FEvents then
    SendMessage(FComPort.FWindow, CM_COMPORT, EV_CTS, 0);
```

```
  if evDSR in FEvents then
    SendMessage(FComPort.FWindow, CM_COMPORT, EV_DSR, 0);
  if evRing in FEvents then
    SendMessage(FComPort.FWindow, CM_COMPORT, EV_RLSD, 0);
  if evRx80Full in FEvents then
    SendMessage(FComPort.FWindow, CM_COMPORT, EV_RX80FULL, 0);
end;

procedure TComThread.DoEvents;
begin
  if evError in FEvents then FComPort.CallError;
  if evRxChar in FEvents then FComPort.CallRxChar;
  if evTxEmpty in FEvents then FComPort.CallTxEmpty;
  if evRing in FEvents then FComPort.CallRing;
  if evCTS in FEvents then FComPort.CallCTSChange;
  if evDSR in FEvents then FComPort.CallDSRChange;
  if evRLSD in FEvents then FComPort.CallRLSDChange;
  if evRx80Full in FEvents then FComPort.CallRx80Full;
end;

{ TComTimeouts }

constructor TComTimeouts.Create;
begin
  inherited Create;
  FReadInterval := -1;
  FWriteTotalM := 100;
  FWriteTotalC := 1000;
end;

procedure TComTimeouts.AssignTo(Dest: TPersistent);
begin
  if Dest is TComTimeouts then
  begin
    with TComTimeouts(Dest) do
    begin
      FReadInterval := Self.ReadInterval;
      FReadTotalM   := Self.ReadTotalMultiplier;
      FReadTotalC   := Self.ReadTotalConstant;
      FWriteTotalM  := Self.WriteTotalMultiplier;
      FWriteTotalC  := Self.WriteTotalConstant;
    end;
  end
  else
```

```
    inherited AssignTo(Dest);
end;

procedure TComTimeouts.SetComPort(const AComPort: TBComPort);
begin
  FComPort := AComPort;
end;

procedure TComTimeouts.SetReadInterval(const Value: Integer);
begin
  if Value <> FReadInterval then
  begin
    FReadInterval := Value;
    FComPort.ApplyTimeouts;
  end;
end;

procedure TComTimeouts.SetReadTotalC(const Value: Integer);
begin
  if Value <> FReadTotalC then
  begin
    FReadTotalC := Value;
    FComPort.ApplyTimeouts;
  end;
end;

procedure TComTimeouts.SetReadTotalM(const Value: Integer);
begin
  if Value <> FReadTotalM then
  begin
    FReadTotalM := Value;
    FComPort.ApplyTimeouts;
  end;
end;

procedure TComTimeouts.SetWriteTotalC(const Value: Integer);
begin
  if Value <> FWriteTotalC then
  begin
    FWriteTotalC := Value;
    FComPort.ApplyTimeouts;
  end;
end;
```

```
procedure TComTimeouts.SetWriteTotalM(const Value: Integer);
begin
  if Value <> FWriteTotalM then
  begin
    FWriteTotalM := Value;
    FComPort.ApplyTimeouts;
  end;
end;

{ TBComPort }

constructor TBComPort.Create(AOwner: TComponent);
begin
  inherited Create(AOwner);
  FComponentStyle := FComponentStyle - [csInheritable];
  FBaudRate := br9600;
  FByteSize := bs8;
  FConnected := False;
  FCTPriority := tpNormal;
  FEvents := [evRxChar, evTxEmpty, evRing, evCTS, evDSR, evRLSD, evError,
            evRx80Full];
  FHandle := INVALID_HANDLE_VALUE;
  FInBufSize := 2048;
  FOutBufSize := 2048;
  FParity := paNone;
  FPort := 'COM2';
  FStopBits := sb1;
  FSyncMethod := smThreadSync;
  FTimeouts := TComTimeouts.Create;
  FTimeouts.SetComPort(Self);
  FUpdate := True;
end;

destructor TBComPort.Destroy;
begin
  Close;
  FTimeouts.Free;
  inherited Destroy;
end;

procedure TBComPort.CreateHandle;
begin
  FHandle := CreateFile(PChar('\\.\' + FPort), GENERIC_READ or
GENERIC_WRITE,
```

```
                    0, nil, OPEN_EXISTING, FILE_FLAG_OVERLAPPED, 0);
  if FHandle = INVALID_HANDLE_VALUE then
  begin
    if GetLastError = ERROR_FILE_NOT_FOUND then
      raise EComPort.Create(CEMess[1])
    else if GetLastError = ERROR_ACCESS_DENIED then
      raise EComPort.Create(CEMess[2]);
  end;
end;

procedure TBComPort.DestroyHandle;
begin
  if FHandle <> INVALID_HANDLE_VALUE then CloseHandle(FHandle);
end;

procedure TBComPort.WindowMethod(var Message: TMessage);
begin
  with Message do
    if Msg = CM_COMPORT then
      try
        if InSendMessage then ReplyMessage(0);
        if FConnected then
          case wParam of
            EV_CTS:      CallCTSChange;
            EV_DSR:      CallDSRChange;
            EV_RING:     CallRing;
            EV_RLSD:     CallRLSDChange;
            EV_RX80FULL: CallRx80Full;
            EV_RXCHAR:   CallRxChar;
            EV_ERR:      CallError;
            EV_TXEMPTY:  CallTxEmpty;
          end;
      except
        Application.HandleException(Self);
      end
    else
      Result := DefWindowProc(FWindow, Msg, wParam, lParam);
end;

procedure TBComPort.BeginUpdate;
begin
  FUpdate := False;
end;
```

```
procedure TBComPort.EndUpdate;
begin
  if not FUpdate then FUpdate := True;
  SetupComPort;
end;

function TBComPort.Open: Boolean;
begin
  if not FConnected then
  begin
    CreateHandle;
    FConnected := True;
    try
      SetupComPort;
    except
      DestroyHandle;
      FConnected := False;
      raise;
    end;
    if (FSyncMethod = smWindowSync) then
      {$IFDEF D6UP}
        {$WARN SYMBOL_DEPRECATED OFF}
      {$ENDIF}
      FWindow := AllocateHWnd(WindowMethod);
      {$IFDEF D6UP}
        {$WARN SYMBOL_DEPRECATED ON}
      {$ENDIF}
    FEventThread := TComThread.Create(Self);
  end;
  Result := FConnected;
end;

function TBComPort.Close: Boolean;
begin
  if FConnected then
  begin
    SetDTR(False);
    SetRTS(False);
    AbortAllAsync;
    FEventThread.Free;
    if FSyncMethod = smWindowSync then
      {$IFDEF D6UP}
        {$WARN SYMBOL_DEPRECATED OFF}
      {$ENDIF}
```

```
      DeallocateHWnd(FWindow);
      {$IFDEF D6UP}
        {$WARN SYMBOL_DEPRECATED ON}
      {$ENDIF}
    DestroyHandle;
    FConnected := False;
  end;
  Result := not FConnected;
end;

procedure TBComPort.ApplyDCB;
const
  CBaudRate: array[TBaudRate] of Integer = (CBR_110, CBR_300, CBR_600,
    CBR_1200, CBR_2400, CBR_4800, CBR_9600, CBR_14400, CBR_19200,
CBR_38400,
    CBR_56000, CBR_57600, CBR_115200, CBR_128000, CBR_256000);
var
  DCB: TDCB;
begin
  if FConnected and FUpdate then
  begin
    FillChar(DCB, SizeOf(TDCB), 0);
    DCB.DCBlength := SizeOf(TDCB);
    DCB.BaudRate := CBaudRate[FBaudRate];
    DCB.ByteSize := Ord(TByteSize(FByteSize)) + 5;
    DCB.Flags := 1 or ($30 and (DTR_CONTROL_ENABLE shl 4))
                 or ($3000 and (RTS_CONTROL_ENABLE shl 12));
    if FParity <> paNone then
      DCB.Flags := DCB.Flags or 2;
    DCB.Parity   := Ord(TParity(FParity));
    DCB.StopBits := Ord(TStopBits(FStopBits));
    DCB.XonChar  := #17;
    DCB.XoffChar := #19;
    if not SetCommState(FHandle, DCB) then
      raise EComPort.Create(CEMess[8]);
  end;
end;

procedure TBComPort.ApplyTimeouts;
var
  Timeouts: TCommTimeouts;

  function MValue(const Value: Integer): DWORD;
  begin
```

```
    if Value < 0 then Result := MAXDWORD else Result := Value;
  end;

begin
  if FConnected and FUpdate then
  begin
    Timeouts.ReadIntervalTimeout := MValue(FTimeouts.ReadInterval);
    Timeouts.ReadTotalTimeoutMultiplier                                    :=
MValue(FTimeouts.ReadTotalMultiplier);
    Timeouts.ReadTotalTimeoutConstant := MValue(FTimeouts.ReadTotalConstant);
    Timeouts.WriteTotalTimeoutMultiplier                                   :=
MValue(FTimeouts.WriteTotalMultiplier);
    Timeouts.WriteTotalTimeoutConstant                                     :=
MValue(FTimeouts.WriteTotalConstant);
    if not SetCommTimeouts(FHandle, Timeouts) then
      raise EComPort.Create(CEMess[9]);
  end;
end;

procedure TBComPort.ApplyBuffer;
begin
  if FConnected and FUpdate then
    if not SetupComm(FHandle, FInBufSize, FOutBufSize) then
      raise EComPort.Create(CEMess[10]);
end;

procedure TBComPort.SetupComPort;
begin
  ApplyBuffer;
  ApplyDCB;
  ApplyTimeouts;
end;

function TBComPort.InBufCount: Integer;
var
  Errors: DWORD;
  ComStat: TComStat;
begin
  if not ClearCommError(FHandle, Errors, @ComStat) then
    raise EComPort.Create(CEMess[11]);
  Result := ComStat.cbInQue;
end;

function TBComPort.OutBufCount: Integer;
```

```
var
  Errors: DWORD;
  ComStat: TComStat;
begin
  if not ClearCommError(FHandle, Errors, @ComStat) then
    raise EComPort.Create(CEMess[11]);
  Result := ComStat.cbOutQue;
end;

function TBComPort.Signals: TComSignals;
var
  Status: DWORD;
begin
  if not GetCommModemStatus(FHandle, Status) then
    raise EComPort.Create(CEMess[12]);
  Result := [];
  if (MS_CTS_ON and Status) <> 0 then Result := Result + [csCTS];
  if (MS_DSR_ON and Status) <> 0 then Result := Result + [csDSR];
  if (MS_RING_ON and Status) <> 0 then Result := Result + [csRing];
  if (MS_RLSD_ON and Status) <> 0 then Result := Result + [csRLSD];
end;

procedure TBComPort.SetDTR(State: Boolean);
var
  Act: DWORD;
begin
  if State then Act := Windows.SETDTR else Act := Windows.CLRDTR;
  if not EscapeCommFunction(FHandle, Act) then
    raise EComPort.Create(CEMess[13]);
end;

procedure TBComPort.SetRTS(State: Boolean);
var
  Act: DWORD;
begin
  if State then Act := Windows.SETRTS else Act := Windows.CLRRTS;
  if not EscapeCommFunction(FHandle, Act) then
    raise EComPort.Create(CEMess[13]);
end;

procedure TBComPort.ClearBuffer(Input, Output: Boolean);
var
  Flag: DWORD;
begin
```

```
  Flag := 0;
  if Input then
    Flag := PURGE_RXCLEAR;
  if Output then
    Flag := Flag or PURGE_TXCLEAR;
  if not PurgeComm(FHandle, Flag) then
    raise EComPort.Create(CEMess[6]);
end;

procedure PrepareAsync(AKind: TOperationKind; const Buffer;
  Count: Integer; AsyncPtr: PAsync);
begin
  with AsyncPtr^ do
  begin
    Kind := AKind;
    if Data <> nil then FreeMem(Data);
    GetMem(Data, Count);
    Move(Buffer, Data^, Count);
    Size := Count;
  end;
end;

function TBComPort.WriteAsync(const Buffer; Count: Integer; var AsyncPtr:
PAsync): Integer;
var
  Success: Boolean;
  BytesTrans: DWORD;
begin
  if AsyncPtr = nil then
    raise EComPort.Create(CEMess[5]);
  PrepareAsync(okWrite, Buffer, Count, AsyncPtr);
  Success := WriteFile(FHandle, Buffer, Count, BytesTrans,
@AsyncPtr^.Overlapped)
    or (GetLastError = ERROR_IO_PENDING);
  if not Success then
    raise EComPort.Create(CEMess[3]);
  Result := BytesTrans;
end;

function TBComPort.Write(const Buffer; Count: Integer): Integer;
var
  AsyncPtr: PAsync;
begin
  InitAsync(AsyncPtr);
```

```
  try
    WriteAsync(Buffer, Count, AsyncPtr);
    Result := WaitForAsync(AsyncPtr);
  finally
    DoneAsync(AsyncPtr);
  end;
end;

function TBComPort.WriteStrAsync(const Str: string; var AsyncPtr: PAsync):
Integer;
begin
  if Length(Str) > 0 then
    Result := WriteAsync(Str[1], Length(Str), AsyncPtr)
  else
    Result := 0;
end;

function TBComPort.WriteStr(const Str: string): Integer;
var
  AsyncPtr: PAsync;
begin
  InitAsync(AsyncPtr);
  try
    WriteStrAsync(Str, AsyncPtr);
    Result := WaitForAsync(AsyncPtr);
  finally
    DoneAsync(AsyncPtr);
  end;
end;

function TBComPort.ReadAsync(var Buffer; Count: Integer; var AsyncPtr: PAsync):
Integer;
var
  Success: Boolean;
  BytesTrans: DWORD;
begin
  if AsyncPtr = nil then
    raise EComPort.Create(CEMess[5]);
  AsyncPtr^.Kind := okRead;
  Success := ReadFile(FHandle, Buffer, Count, BytesTrans, @AsyncPtr^.Overlapped)
    or (GetLastError = ERROR_IO_PENDING);
  if not Success then
    raise EComPort.Create(CEMess[4]);
  Result := BytesTrans;
```

```
end;

function TBComPort.Read(var Buffer; Count: Integer): Integer;
var
  AsyncPtr: PAsync;
begin
  InitAsync(AsyncPtr);
  try
    ReadAsync(Buffer, Count, AsyncPtr);
    Result := WaitForAsync(AsyncPtr);
  finally
    DoneAsync(AsyncPtr);
  end;
end;

function TBComPort.ReadStrAsync(var Str: string; Count: Integer; var AsyncPtr:
PAsync): Integer;
begin
  SetLength(Str, Count);
  if Count > 0 then
    Result := ReadAsync(Str[1], Count, AsyncPtr)
  else
    Result := 0;
end;

function TBComPort.ReadStr(var Str: string; Count: Integer): Integer;
var
  AsyncPtr: PAsync;
begin
  InitAsync(AsyncPtr);
  try
    ReadStrAsync(Str, Count, AsyncPtr);
    Result := WaitForAsync(AsyncPtr);
    SetLength(Str, Result);
  finally
    DoneAsync(AsyncPtr);
  end;
end;

function ErrorCode(AsyncPtr: PAsync): Integer;
begin
  if AsyncPtr^.Kind = okRead then Result := 4 else Result := 3;
end;
```

```
function TBComPort.WaitForAsync(var AsyncPtr: PAsync): Integer;
var
  BytesTrans, Signaled: DWORD;
  Success: Boolean;
begin
  if AsyncPtr = nil then
    raise EComPort.Create(CEMess[5]);
  Signaled := WaitForSingleObject(AsyncPtr^.Overlapped.hEvent, INFINITE);
  Success := (Signaled = WAIT_OBJECT_0) and
      (GetOverlappedResult(FHandle, AsyncPtr^.Overlapped, BytesTrans, False));
  if not Success then
    raise EComPort.Create(CEMess[ErrorCode(AsyncPtr)]);
  Result := BytesTrans;
end;

procedure TBComPort.AbortAllAsync;
begin
  if not PurgeComm(FHandle, PURGE_TXABORT or PURGE_RXABORT) then
    raise EComPort.Create(CEMess[6]);
end;

function TBComPort.IsAsyncCompleted(AsyncPtr: PAsync): Boolean;
var
  BytesTrans: DWORD;
begin
  if AsyncPtr = nil then
    raise EComPort.Create(CEMess[5]);
  Result := GetOverlappedResult(FHandle, AsyncPtr^.Overlapped, BytesTrans,
False);
  if not Result then
    if (GetLastError <> ERROR_IO_PENDING) and (GetLastError <>
ERROR_IO_INCOMPLETE) then
      raise EComPort.Create(CEMess[7]);
end;

procedure TBComPort.CallCTSChange;
begin
  if Assigned(FOnCTSChange) then FOnCTSChange(Self, csCTS in Signals);
end;

procedure TBComPort.CallDSRChange;
begin
  if Assigned(FOnDSRChange) then FOnDSRChange(Self, csDSR in Signals);
end;
```

```
procedure TBComPort.CallRLSDChange;
begin
  if Assigned(FOnRLSDChange) then FOnRLSDChange(Self, csRLSD in Signals);
end;

procedure TBComPort.CallError;
var
  Errs: TComErrors;
  Errors: DWORD;
  ComStat: TComStat;
begin
  if not ClearCommError(FHandle, Errors, @ComStat) then
    raise EComPort.Create(CEMess[11]);
  Errs := [];
  if (CE_FRAME and Errors) <> 0 then Errs := Errs + [ceFrame];
  if ((CE_RXPARITY and Errors) <> 0) and (FParity <> paNone) then
    Errs := Errs + [ceRxParity];
  if (CE_OVERRUN and Errors) <> 0 then Errs := Errs + [ceOverrun];
  if (CE_RXOVER and Errors) <> 0 then Errs := Errs + [ceRxOver];
  if (CE_TXFULL and Errors) <> 0 then Errs := Errs + [ceTxFull];
  if (CE_BREAK and Errors) <> 0 then Errs := Errs + [ceBreak];
  if (CE_IOE and Errors) <> 0 then Errs := Errs + [ceIO];
  if (CE_MODE and Errors) <> 0 then Errs := Errs + [ceMode];
  if (Errs <> []) and Assigned(FOnError) then FOnError(Self, Errs);
end;

procedure TBComPort.CallRing;
begin
  if Assigned(FOnRing) then FOnRing(Self);
end;

procedure TBComPort.CallRx80Full;
begin
  if Assigned(FOnRx80Full) then FOnRx80Full(Self);
end;

procedure TBComPort.CallRxChar;
var
  Count: Integer;
begin
  Count := InBufCount;
  if (Count > 0) and Assigned(FOnRxChar) then FOnRxChar(Self, Count);
end;
```

```
procedure TBComPort.CallTxEmpty;
begin
  if Assigned(FOnTxEmpty) then FOnTxEmpty(Self);
end;

procedure TBComPort.SetBaudRate(const Value: TBaudRate);
begin
  if Value <> FBaudRate then
  begin
    FBaudRate := Value;
    ApplyDCB;
  end;
end;

procedure TBComPort.SetByteSize(const Value: TByteSize);
begin
  if Value <> FByteSize then
  begin
    FByteSize := Value;
    ApplyDCB;
  end;
end;

procedure TBComPort.SetParity(const Value: TParity);
begin
  if Value <> FParity then
  begin
    FParity := Value;
    ApplyDCB;
  end;
end;

procedure TBComPort.SetPort(const Value: String);
begin
  if FConnected then
    raise EComPort.Create(CEMess[14])
  else
    if Value <> FPort then FPort := Value;
end;

procedure TBComPort.SetStopBits(const Value: TStopBits);
begin
  if Value <> FStopBits then
```

```
  begin
    FStopBits := Value;
    ApplyDCB;
  end;
end;

procedure TBComPort.SetSyncMethod(const Value: TSyncMethod);
begin
  if Value <> FSyncMethod then
  begin
    if FConnected then
      raise EComPort.Create(CEMess[14])
    else
      FSyncMethod := Value;
  end;
end;

procedure TBComPort.SetCTPriority(const Value: TThreadPriority);
begin
  if Value <> FCTPriority then
  begin
    if FConnected then
      raise EComPort.Create(CEMess[14])
    else
      FCTPriority := Value;
  end;
end;

procedure TBComPort.SetInBufSize(const Value: Integer);
begin
  if Value <> FInBufSize then
  begin
    FInBufSize := Value;
    if (FInBufSize mod 2) = 1 then Dec(FInBufSize);
    ApplyBuffer;
  end;
end;

procedure TBComPort.SetOutBufSize(const Value: Integer);
begin
  if Value <> FOutBufSize then
  begin
    FOutBufSize := Value;
    if (FOutBufSize mod 2) = 1 then Dec(FOutBufSize);
```

```
    ApplyBuffer;
  end;
end;

procedure TBComPort.SetTimeouts(const Value: TComTimeouts);
begin
  FTimeouts.Assign(Value);
  ApplyTimeouts;
end;

procedure InitAsync(var AsyncPtr: PAsync);
begin
  New(AsyncPtr);
  with AsyncPtr^ do
  begin
    FillChar(Overlapped, SizeOf(TOverlapped), 0);
    Overlapped.hEvent := CreateEvent(nil, True, True, nil);
    Data := nil;
    Size := 0;
  end;
end;

procedure DoneAsync(var AsyncPtr: PAsync);
begin
  with AsyncPtr^ do
  begin
    CloseHandle(Overlapped.hEvent);
    if Data <> nil then FreeMem(Data);
  end;
  Dispose(AsyncPtr);
  AsyncPtr := nil;
end;

procedure EnumComPorts(Ports: TStrings);
var
  KeyHandle: HKEY;
  ErrCode, Index: Integer;
  ValueName, Data: String;
  ValueLen, DataLen, ValueType: DWORD;
  TmpPorts: TStringList;
begin
  ErrCode                 :=                 RegOpenKeyEx(HKEY_LOCAL_MACHINE,
'HARDWARE\DEVICEMAP\SERIALCOMM',
                    0, KEY_READ, KeyHandle);
```

```
  if ErrCode <> ERROR_SUCCESS then
    raise EComPort.Create(CEMess[15]);
  TmpPorts := TStringList.Create;
  try
    Index := 0;
    repeat
      ValueLen := 256;
      DataLen := 256;
      SetLength(ValueName, ValueLen);
      SetLength(Data, DataLen);
      ErrCode := RegEnumValue(KeyHandle, Index, PChar(ValueName),
                              {$IFDEF VER120}
                              Cardinal(ValueLen),
                              {$ELSE}
                              ValueLen,
                              {$ENDIF}
                              nil, @ValueType, PByte(PChar(Data)), @DataLen);
      if ErrCode = ERROR_SUCCESS then
      begin
        SetLength(Data, DataLen);
        TmpPorts.Add(Data);
        Inc(Index);
      end
      else
        if ErrCode <> ERROR_NO_MORE_ITEMS then
          raise EComPort.Create(CEMess[15]);
    until (ErrCode <> ERROR_SUCCESS) ;
    TmpPorts.Sort;
    Ports.Assign(TmpPorts);
  finally
    RegCloseKey(KeyHandle);
    TmpPorts.Free;
  end;
end;

procedure Register;
begin
  RegisterComponents('Samples', [TBComPort]);
end;

end.
```

Приложение Д. Принципиальная схема платы клндуктометра